DID YOU KNOW?

We still have the very best PD library for the Atari Classic

There are over 400 disks available

Many disks are only £1.50 each

Collections and special sets are even cheaper

EVERY PURCHASE OF A PD DISK HELPS TO SUPPORT THE MAGAZINE

DID YOU REALISE?

We still have the entire ST library available

There are over 1,000 disks

Disks are only £1 each

We will send you details on request

KEEP SUPPORTING US
AND WE WILL KEEP SUPPORTING YOU

Page 6 Publishing's NEW

ATARI USER

The Resource for the ATARI CLASSIC and the ATARI ST

Issue 84 - May/June 1998 £2.50

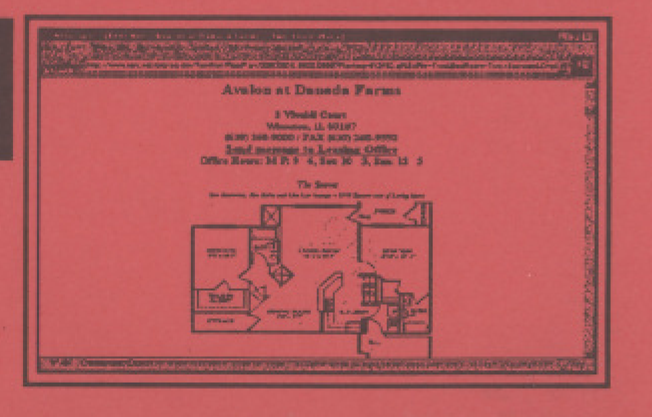
FOR THE ATARI CLASSIC



- ***THE SEARCH IS ON***
  Search and sort routines
- ***USEFUL USR***
  Crack the connection between Basic and machine code
- ***CD COLLECTION DATABASE***
  The next best thing to a CD cabinet!

THE WIDER SCENE

FINDING A HOME ON THE INTERNET



PLUS ... WORM IN PARADISE ... HAPPY TYPER ... SPARTADOS ... MAILBAG ... AND MORE!

This issue's ....

Thanks

Les Ellingham puts it all together and fills up the gaps but the real thanks goes to the following who made this issue possible

Sandy Ellingham who takes care of all the office work, advertising and mail order

For their contributions this issue

Raphael Espino
Allan Palmer
Dennis Hedges
James Austin
Eddie Jones
Kevin Cooke
Paul Herbert
John Foskett
James Matthrick
Joel Goodwin
Dennis Fogerty
Austin Hillman
Daniel Yelland
M. Tomlin
John S Davison

APOLOGIES

I am still extremely poor in acknowledging contributions so I apologise to everyone who has sent in stuff and thought it has gone through the wormhole. The intention to reply to everyone is there but the time seems to drift by. If you have not heard, thank you and keep watching the mag, you might be surprised.

HOW IT'S DONE

PAGE 6 shows just what you can do with your Atari. NEW ATARI USER has always been created entirely with Atari equipment, initially on the XL but more lately with a Mega ST and other stuff, who needs PC's or Macs! Hardware includes a Mega ST2 (upgraded to 4Mb), SM125 Monitor, Supra 30Mb Hard Disk, a HP Laserjet III, Citizen 124D printer, Philips CM8833 monitor, 130XE, a couple of 1050 disk drives, 850 interface, NEC 8023 printer. Principal software used is Protext and Fleet Street Publisher 3.0. Other software includes Kermit, TariTalk, Turbo Basic and various custom written programs on the XL/XE. Articles submitted on XL/XE disks are transferred across to the ST via TARITALK. Programs are coded on the XE and printed out directly for pasting in after the typesetting is completed. All major editing is done with Protext and pages are laid out with Fleet Street Publisher. Each page is output directly from Fleet Street to a HP Laserjet III which produces finished pages exactly as you see them. All that is left is to drop in the listings and photos.

Well, it's not quite as easy as that but you get the idea!

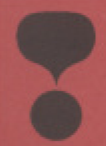
Inspiration

It seems such a long time between issues that I often have to look up what I was listening to last time. Quite often I think that I have been listening to a particular CD for ages only to find it wasn't on the playlist last time. Such is the case with the CD playing right now by Walela. This group comprises of Rita Coolidge, her sister Priscilla and Laura Satterfield who I think is Priscilla's daughter. It is (naturally!) of Native American influence though no more so than the albums by Buffy St. Marie of the sixties and seventies. Rita Coolidge is of Cherokee descent and has recently decided to embrace her roots more deeply. The result is a superb collection of modern songs with just the right Native American influence. I bet I'm the only person to have discovered this, after all how many people look in the Native American section of the World Music rack in their local store? I can't understand why this is not promoted in the mainstream music section. Also being played recently is a compilation including tracks by Jim Page (no, not Jimmy), Joanne Shenandoah, Ellen Klaver, Larry Long and Julie Robbins. Bet you have never heard of any of them. Write and tell me if you have.

CONTRIBUTIONS

Without contributions from its readers, NEW ATARI USER would not be possible. PAGE 6 welcomes and encourages its readers to submit, articles, programs and reviews for publication. Programs must be submitted on disk or cassette, articles should wherever possible be submitted as text files on disk. We seek to encourage your participation and do not have strict rules for submissions. If something interests you, write a program or article and submit it!

COPYRIGHT



Editorial address: P.O. Box 54, Stafford, ST16 1DR, ENGLAND Tel. 01785 241153
Editor & Publisher: Les Ellingham - *Advertising:* Sandy Ellingham
Page layout by PAGE 6 - Printed by Dolphin Press, Fife, Scotland 01592 771652
NEW ATARI USER is published bi-monthly on the last Thursday of the month prior to cover date

PAGE 6 PUBLISHING's NEW ATARI USER

'The Magazine for the Dedicated Atari User'

ISSN No. 0958-7705

CONTENTS

Issue 84 - May/June 1998

REGULARS

ONLY A SHORT WHILE TO THE NEXT COPY DATE - PLEASE CONTRIBUTE NOW!

PROGRAMMING

FEATURES

COPY DEADLINE FOR NEXT ISSUE - 12th JULY

SUBSCRIPTIONS

MAGAZINE ONLY

Annual subscription rates (6 issues)

UK	£15.00
Europe (Air Mail)	£17.00
Elsewhere (Surface)	£17.00
Elsewhere (Air Mail)	£23.00

Overseas rates reflect only the difference in postal costs

DISK SUBSCRIPTION

A disk containing all of the 8-bit programs from each issue of NEW ATARI USER is available either separately or on subscription. Single price £2.95 per disk, a disk subscription saves you almost £8 a year. Subscription rates (6 issues)

UK	£25.00
Europe	£32.00
Elsewhere (sea)	£32.00
Elsewhere (Air)	£42.00

Please make cheques payable to PAGE 6 PUBLISHING and send to PAGE 6 Publishing, P.O. Box 54, Stafford, ST16 1DR

Editorial

For one of the few times in the past ten years or so I can't readily think of something to rattle on about in the editorial so I guess I'll go to the old standby of asking for your contributions. In fact this not really a standby but an important topic at the moment for if you don't contribute something we will soon be struggling to fill the magazine. We're okay for the next couple of issues but there is very little in reserve and I need to think far ahead.

Whenever I make these pleas for contributions the old faithfuls such as John Foskett always respond with another disk or two full of articles and programs and, whilst these are more than appreciated, we can't keep relying on the same contributors time and time again. Apart from anything else these people will probably dry up one day so don't rely on others to make the contributions, try and think of something you can do. Of course not everyone can write programs or compose articles but anyone can write a letter and that is a good starting point. A full Mailbag is a solid foundation for any issue and you can begin by responding to any topics that interest you in this issue's Mailbag. If there is nothing much of interest then write to us about something you are interested in, it doesn't even have to be about Atari related computing, as the recent interest in the Internet has shown.

If you want to do more than just write a letter, then a good idea is to take your favourite program (a PD program is probably best since these are readily available) and do a comprehensive write up on it. If you use a program regularly then you should know it inside out by now. Share that knowledge with others. Don't be afraid to do an overview on similar types of programs, for example comparing the various word processors that are available. How about taking a PD program and writing about what isn't in it? Tell readers about what you would like to see in your dream program and maybe someone will be inspired to write it.

So what other ideas are there?

- Send in a favourite programming routine you use often and tell us how it works.
- Polish off a program you have written and send it in.
- Find good programs from other publications, as Dennis Fogerty did this issue.
- Find an article somewhere that you think could be written better and make it interesting for us all.
- Dig out some good PD programs we don't have and send them in to be used as disk bonuses.

And this is only a few of the ways you can help!

THIS IS IMPORTANT

I have said many times that this is your magazine, full of your contributions and you must stay committed for us to survive. With the pressure of work in trying to keep the mortgage paid I simply don't have the time to fill in huge sections of the magazine as I may have done several years ago so I am relying on you more than ever to make my job a lot easier, and to make New Atari User as interesting as it has always been.

You now it makes sense. Please start thinking now and writing tomorrow! Once you start it becomes a lot easier and when you see your name in print you will be inspired to contribute again. That's our future.

Les Ellingham



PROGRAMMING

3D RIPPLE

Raphael Espino presents this neat little program for a great graphic design

This program creates a 3D ripple surface which serves no purpose except that it is interesting to look at once it is complete. It does take a while to run so you will need some patience but the result is well worth it. I suggest that you run the program with Turbo Basic which will speed up the drawing or even try compiling it with the Turbo compiler. Remember to save a Basic version though as you might want to experiment with the parameters.

```
NG 1 REM
SJ 2 REM |  3D RIPPLE BY RAPHAEL ESPINO |
NI 3 REM
IO 10 GRAPHICS 24:COLOR 1:XM=160:ZM=150:Y
   M=130:ANG=20:AN=3.14159/180*ANG:RAD :S
   T=SIN(AN):CT=COS(AN)
IK 20 PXH=192:PXW=320:CX=160:CY=50:CM=CT*
   YM:TRAP 100:SW=2*XM/PXW:SH=(YM+CY)/PXH
AH 30 FOR X=-XM TO 0 STEP 1:FOR Y=-ZM TO
   ZM STEP 4
JO 40 V=SQR(X*X+Y*Y)/10:SN=SIN(V)/V:YP=Y*
   ST+CM*SN
BH 50 IF Y=-ZM THEN MINY=YP:MAXY=YP
XD 60 IF YP>MAXY THEN MAXY=YP:PLOT CX+X/S
   W,(PXH-2)-(YP/SH+CY):PLOT (PXW-1)-(CX+
   X/SW),(PXH-2)-(YP/SH+CY)
TG 70 IF YP<MINY THEN MINY=YP:PLOT CX+X/S
   W,(PXH-2)-(YP/SH+CY):PLOT (PXW-1)-(CX+
   X/SW),(PXH-2)-(YP/SH+CY)
ZX 80 NEXT Y:NEXT X:POKE 77,128
UK 90 GOTO 90
NU 100 TRAP 100:GOTO 80
```

MODIFYING THE PROGRAM

If you want to try and modify the program note that the main variables are:

XM	width of the surface
YM	height of the "bumps"
ZM	apparent depth of view
ANG	angle of view (in degrees)
PXH	height of screen in pixels
PXW	width of screen in pixels
CX	centre of shape horizontally
CY	centre of shape vertically

To use a different graphics mode change the GRAPHICS command in line 10, and variables PXH, PXW, CX and CY to suit your new mode. For example, mode 7+16 would be PXH=96 PXW=160 CX=80 CY=25. CY should be roughly one quarter of PXH and CX half of PXW. You should also change the values in the STEP commands in line 30 by doubling them each time the screen size is halved.

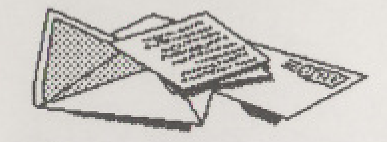
To change the shape's position on the screen then change CX and CY and to change it's height and width then change PXH and PXW.

Try also changing the following lines.

Line 30 - increase the numbers in the STEP commands to speed up the drawing. This will make the drawing "blockier" however.

Line 40 - the second equation (SIN(V)/V) is the one that determines the shape of the surface. Also try SIN(V)^2/V and SIN(V)*COS(V)/V for different shapes. ●

Mailbag



This issue's Mailbag conducted by Les Ellingham

INTERNET SUCCESS!

After detailing my experiences on the Internet last issue, Allan Palmer came to my rescue and provided a couple of pages of Web addresses relating to Leonard Peltier and other 'links' of interest. He obviously has the experience to find what I could not. Some of his comments will be of general interest so here are a few things he had to say:

"Having dabbled with the Internet and creating a simple home page when I first acquired a PC back in 1995, I'm now a confessed cybernaut (or should that be cybernut?). Part of my spare time is spent in maintaining the Web pages relating to The Cavaliers - Basingstoke's award-winning youth marching showband (of which my daughters are members), and also to maintaining the Web pages of the British Youth Band Association (BYBA) - the national organisation which co-ordinates and promotes youth banding. In case anyone's interested, the URLs for these pages are http://www.the-cavaliers.org.uk/ and http://www.byba.org.uk/ respectively.

Your editorial recounted your attempts to research Leonard Peltier on the Internet and you were surprised that using the Yahoo search engine you only retrieved a handful of hits. This may be down to a couple of things - firstly, the content of the Net continues to grow exponentially, keeping track of it is more than a full time job; secondly, web pages are indexed by tags within the HTML code which list relevant keywords for search engines to use, if a word or phrase you're interested in isn't included in the tag, it won't be indexed. There are some search engines which build their indexes by scanning the web and the actual content of pages, but of course they've got to find all the pages.

I did a quick search via Webcrawler and came up with a number of hits concerning Leonard Peltier and enclose a print out of various addresses which may be of use next time you go cyber-surfing. As opposed to a Cyber Cafe, have you tried your local library? Ours provides Internet access, I understand, of some form."

❢ *Allan's list of addresses was superb and has enabled me to get much more value from my second fiver. As to local libraries, some people are lucky, others not, depending on the local authority. Staffordshire libraries have no public Internet access of any kind as they "are awaiting a policy decision". I do know that the main library in Southend offers free Internet access, all you have to do is book your hour in advance. Also the library in the little village of Street in Somerset offers public access, although I don't know if you have to pay. If they can do it why can't all libraries?*

It would be an excellent idea if we could put together an article on public Internet access with the help of our readers. If any of you know of Cyber Cafes in your area, find out what they charge and send us the details together with their postal address. If you use the Internet at your local library let us have details of what they charge and what their policy is, for instance do you have to be a member of that particular library or is access open to anybody? Is there such a thing as 'Libraries On-line' on the Internet that will give details of all the libraries that are connected up? Here's a good chance for some of you to contribute to an article, even if you can't write the article yourself (although a volunteer would be more than welcome!). The Internet really is a fabulous tool for research and it would be good if those of us without PCs or modems could get to use it a little more easily.

TRANSFER REQUEST

Dennis Hedges from Southampton has been having a few problems transferring data between the Atari and his PC but has worked out some of the problems as he explains: "With regard to Philip Brown's problems with the G.M. Transfer cable, I had the same trouble of no interaction between the PC and Atari. Having printed out the 36 pages of information from the file Readme.txt on the disk, and anything else I could find to read, I must say that Nick Kennedy really does explain everything very well, although I did not understand a lot of the technical parts.

Having read it all, I still could not get the Atari to recognise my PC. Anyway I did manage to crack it thus: Get SIO2PC up and running on the PC, then load MYDOS453.ATR on to the virtual Drive One. Then press A on the PC keyboard to get the timing screen up. Press T which toggles the sense of command line input. Switch on the Atari and the command line at the bottom of the program should change to DEV 31 or 4F. Type DOS on the Atari and you should get the DOS program on the monitor. I did experiment with some of the timings, but this did not seem to make much difference, only on UART.

There seems to be a lot more to transferring games and files than is made out but I hope that this will help someone."

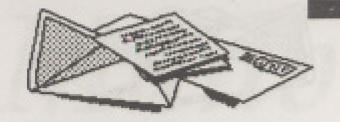
❢ *Thanks Dennis, I have to admit that I didn't understand much of it, but I guess if you've got the program up on the monitor in front of you it will make sense. I hope that Philip is now up and running and maybe some others will be helped as well.*

MORE INTERNET

James Austin's letter arrived just a couple of days after I had completed the last issue so refers to items raised in Issue 82. It is still interesting so let's hear from James:

"First off, regarding the letter from H.S. Wood, could he, or anyone else, explain what a P.I.C. is? Presumably they are something to do with PC's. As regards connecting an Atari to a PC, I don't currently own a PC and don't intend to get one in the near future, but I do find it interesting to read about, as long as the subject matter does not get too technical, because it then goes way over the head of users such as myself who do not own a PC. It's okay as long as it doesn't take up too much of an issue and so overshadow other topics. It hasn't happened yet but I feel that we should be careful not to let this happen.

On a similar note, I have found reading about the Internet, in John S Davison's 'Journey Into Cyberspace' very interesting. It is a pity that it is so expensive though. I have only been on the Internet once or twice at school, when the teachers aren't looking or when it has accidentally been left turned on, which isn't often, but have managed to find several ATARI-specific Web sites, including an ATARI 8-Bit web site at the University of Kent at Canterbury (presumably there are similar web sites at other universities?), also a



"Digital Antic/Digital Analog" web site where all the articles/programs from the magazines can be found and downloaded. There is also a really interesting site where screen shots and information on all the classic Atari games can be downloaded and viewed. There also appears to be a lot of information on the Net about new ATARI commercial games from the States. I never had the time to write down any of the Web site addresses for these but I found them pretty easily using the Netscape search facility, so perhaps John or someone else with access can have a look around for some of these sites and maybe send in information about them, maybe also sending in any downloaded articles/information that might be of interest to readers. Perhaps a regular section could be set up in the magazine, in addition to John S Davison's column, where this stuff could be sent in and published for everyone who can't get Internet access.

Before I close, a couple of cries for help. Firstly, does anyone out there own a 1029? If you do then would anyone care to test a new piece of software that I have written for it? If so PLEASE contact me at the address below. It will come with a manual fully explaining how to use it and detailing all of its features. I'm asking this because I really need someone impartial to test it out for me to see if it's worth me completing it and maybe sending it in to PAGE 6. Also, does anyone know where I can get a full working version of Daisy Dot III (i.e. the one that allows you to use multiple fonts in a document, has SpartaDos X support, etc.)? If you do then please write to me at the address given."

Thanks for some interesting points James. Your comments about using the Internet at school seem to echo other school's treatment of the Internet. At my son's school they got connected up several months ago via a special deal with the local cable company which effectively cost them peanuts. When they first talked about it they said that they could probably offer general Internet access at around 80p at hour, and actually make a profit from it. It has turned out however that use has been severely restricted so that pupils can only use the Internet during school time in connection with school projects only in the presence of teacher. What a total waste of such an important research tool! They are many responsible children (my son included) who would love to use the Internet in their own time, especially at low cost, who are being denied such use because the school can't work out how to use and supervise the system properly. They are basically scared stiff that pupils will use the Internet to access pornographic material and because they don't know how to control this they ban all pupils. Not only is that copping out from tackling a problem, it is also an insult to the majority of responsible children who could expand their education enormously in their own time through accessing information on subjects that they are genuinely interested in. Maybe one day it will get sorted, but I must confess that I have little regard for the majority of teachers in State schools, who seem to be interested in little beyond their immediate responsibilities.

As regards James's requests you can write to James Austin at 19, Clive Road, Grove Park Estate, Bobbing, Nr. Sittingbourne, Kent ME10 1PJ.

BIND IT

Eddie Jones has a couple of contact addresses and an idea for a binder you can make to hold your copies of NAU.

"The following telephone numbers may be of use to owners of EPSON equipment - EPSON 0800 289622, department required is by number selection. If you require spares or optional add-on parts they will refer you to their distributors. Try also Micro Partners (Wembley) on 0800 253550. They are a Mail order company and their prices are cheaper than any of the Dixon Group.

If you want to have a go at making your own binders for NAU, purchase an A5 REXEL BUDGET Binder No 13428 for approximately £1. It is a perfect size and soft backed and it will hold 6 issues. Also buy a packet of ELASTIC CORD 1½/2mm diameter, about 5 metres long. It will cost you about another £1 from Haberdashery Shops, Sainsbury, Tesco etc.

Remove the metal sprung clip using a pair of pliers that will crimp the inner end of the rivet and allow you to push it out. The elastic is cut into approximately 15cm lengths but you may like to experiment with the length, so you can get the tension you prefer. At the top of the spine on the outside make a mark 15mm from the top and divide the width into seven spaces giving 6 marks where the 6 pieces of elastic will be glued. Use 15mm of the elastic unstretched to glue here and then do the remainder. At the bottom end of the spine repeat as the top, once the top glue has set. You will find the binder will bend, but when the magazines are fitted all will be well. The exposed glued ends can be covered, I used sticky tape, black at the bottom and red at the top.

If you cannot get the soft backed type of binder then use the hard backed variety. These are wider approximately, 45mm more. Somebody may have a better idea, if so let's have it."

OTHER COMPUTERS

Regular contributor Kevin Cooke has quite a number of things to talk about: "I have just received the latest issue of N.A.U. and must say that I was most impressed. As usual, it was full of interesting articles for people of all levels of knowledge, making an excellent read.

I'm now attending university at Bristol and as such, I now have an e-mail address by which other Atari users can contact me: KJ-COOKE@WP-G.UWE.AC.UK. There is only so much help I can give people as I don't have my Atari with me in Bristol (for reasons outlined in the rest of my letter) but I will do my best to answer any questions relating to general Atari use, and am quite willing to have a chat with anyone. Please add my e-mail address to the NAU Internet contact list.

My time at university also gives me free access to the Internet, so I was interested to read of your own experiences with the World Wide Web (WWW). I have tried to use the WWW in the past to do some serious research for university projects, usually with limited success. For example, recently I have been researching into a phenomenon known as "Sick Building Syndrome" (S.B.S.). However, when typing "SBS" into the search engine, I received information on loads of subjects not in any way related to this phenomena (such as the Special Boat Service and something about nursing!). The stuff that WAS relevant to the subject was mostly written by companies trying to sell items to eliminate the problem.

Another danger with the Internet is taking everything that you read on it at face value. One of the biggest problems with the Internet is that anyone can create webpages with biased, prejudiced views and present it as fact.





A lot of the so called "facts" written by "experts" are extremely hard to verify, with pages disappearing from the Net as quickly as they appeared. I'm also surprised at how many people use the Internet, considering how long it takes to load the pages you are looking for. It's little surprise that it's been dubbed "The World Wide Wait"!

In issue #82's mailbag, Joel Goodwin suggested that other readers may like to hear of Atari enthusiasts' experiences on other computers. Recently, I've been using PCs and an Amiga rather a lot. Here's why:

I've recently sold my Atari ST and used the money to buy an Amiga. OK, I hear cries of outrage and surprise from the Atari community, but let me explain myself. I bought the ST as a step up from the 8-bit for desk-top publishing and, although it satisfied me at first, I soon became annoyed at how the programs I was using kept crashing, partly due to bugs in the software but mainly due to my machine only having 1 Meg and no hard drive. The cure was simple - buy more memory and a HD. But at what cost? It turned out that if I sold my ST and software (for £70), I could buy an Amiga 1200 with hard drive, 4 megs of memory, an extra floppy drive, and loads of software for only £100 more. If you look at the prices that Atari users generally want for memory and/or systems with hard drives second hand, you'll be looking at a lot more money than one hundred pounds (or at least you would have been seven months ago when I bought the Amiga). At the end of the day, money had to talk, and with the collapse of Atari, I saw little loyalty left to a company who had shown me none. And, as much as I liked the ST, I have to admit that the Amiga is FAR superior. For example, filenames are not limited to 8 characters (in fact they can be about 20 characters long). The desktop is nicer to use, especially with the various shareware improvements available, and I can't remember any software ever having crashed (unlike the ST, where even half-meg games would crash for no apparent reason). The joystick/mouse ports are in a much easier to access position, the keyboard feels nicer (I never could get used to the ST keyboard!) and the serious software is cheaper and still being supported/improved. So is my love affair with Atari over? No! My comments may seem controversial and it's true, I don't have my 8-bit at university with me (because of space limitations and the fact that my Amiga can save text files straight onto a 3.5" disk and load them into a PC, which is essential for my university course) but you just can't beat the old 8-bit for easy to use, easy to adapt fun! I have a feeling I may buy an ST again when I have more money but for now, I'll be using my Amiga and Atari 8-bit with satisfaction.

It's interesting to hear people criticise the 8-bit and 16-bit computers and rave about their powerful PCs, only to later prove that they don't even know how to use a fraction of the functions available on their PCs. For example, a student in my class made a big deal of the new Pentium computer she had bought when she started the degree, yet for six months when she thought she was saving her work onto a blank disk, she found she had been saving it to the hard drive! Therefore, all the work she thought she had saved to one floppy disk had actually been spread amongst the numerous computers she had used at university, and the one she had used at home!

In another instance, when I was doing a group project for one of my degree modules, I gave one member of the group my work on a disk, along with a printed copy, so she could paste it together with each group member's work during the weekend to create a finished essay. When the girl got home and tried to load it into her Pentium, she was shocked to find that there didn't appear to be any file on the disk. After panicking, she ended up typing in all of the work from the printed copy (about 5 pages). After the weekend, she complained to me that the disk was empty. It turned out that her word processor was only looking for files with an extender of .TXT and because my file was saved with a .WPF extender, it hadn't showed up on the directory! All she had to do was change the type of files that would be listed in the directory from *.TXT to *.*, which she didn't know how to do!

I'm also amazed at the number of PC owners who can't even copy or format a disk! And would you believe that you can't swap floppy disks whilst using programs in Windows (or "Win-doze" as it's been called!) because it corrupts the data table on the disks, effectively losing everything on the disk!

I think that these examples go to show that it's not just the capability of the computer that is important, but also the capability of the user!"

❢ *Interesting comments, Kevin, which I think shows the difference between the computer enthusiast and the computer user. Those of us who came into computing via the Atari generally have a greater understanding about how things work and so will benefit from using any kind of computer. The vast majority of PC users don't understand in the slightest what is going on which is why you get all these 'consultants' ripping people off with training schemes and the like. It is unbelievable the amount of money that schools and businesses waste because the people working there can't read a manual. It also astonishes me that people need to go on training courses to learn how to use a computer, I don't mean how to write programs, but just so that they can learn how to press keys! How many of you reading this had to go on a course to learn to use the Atari?*

NEVER TOO LATE!

Paul Herbert starts off by letting us know that it is never too late to start writing to Mailbag: "After nine years of reading Page 6/NAU, I've finally gotten off my backside and written in!

First of all, some Transdisk problems. Feud and Head Over Heels both seem to load in one chunk, making it impossible to transfer in stages, and the programs are too big to hold in my XE's 64k capacity (with Transdisk loaded). Shame, because they are two of my favourites, especially H/H. Red Max, on the other hand, loads in enough stages to successfully transfer to disk (514 sectors! Phew!), but doesn't want to load from disk. It's not my copy of the game, because I loaded it from tape with success (half an hour later ..... those were the days), so something went wrong in the transfer. Is there anyone who has Transdisk-ed these games successfully who can help?

Raking through my old tape collection I came across three cassettes that I had long forgotten about, and I would be curious to know if anyone else has any copies of them. They are issues 3, 4 and 5 of a cassette-magazine called Atari Computing, dated 1984. They have various games, utilities, and reviews (including a review of Blue Max when it first came out!) on them, plus a three part text adventure (written entirely in Basic) called the Keys of Time. If it helps ring any bells, the address was 1



Golden Square, London W1R 3AB (Argus Press Software). Does anyone know what happened to Argus Press Software? If the copyright status permits (it says (c)APS on the tapes) I would be more than willing to transfer these onto disk and to send them in for Bonus Progs or Public Domain - we all know how short on these Les is.

On the subject of potential Bonus Programs, I have a book of type-in games called Games For Your Atari (Virgin, 1983), many of which I have typed in. Needless to say, some of them are pretty awful ("Compliment Generator" - random phrases to boost your ego!), but there are some half decent type-in games lurking around, and again, copyright permitting, I would be happy to send them in and share them with other users if possible.

Finally, I discovered the real uses of demos. Recently, a friend came round my house and enquired as to the nature of my computer, given that it clearly wasn't a PC - surely they are the only types of computer in the world?? Wrong again. After telling him that it was an 8-bit Atari and having received the usual stuck-up noises that I have come to expect, I asked him for a few minutes of his time, and loaded up the "Polish Demos 2" disk (#161 in the Page 6 PD library if anyone's interested). This has a collection of demos, but I went straight to the one with the animation of a dancer moving to the theme tune from Draconus AND two simultaneous scroll texts (approx.400k compressed onto a standard 130k disk). After he ate his words (reluctantly), I informed him that the processing technology was older than me (being a fledgling 20 years - young for a human, but in computer terms that's about 40 generations). Ooaar! They don't make computators loike they used to!"

I don't think anyone ever read Atari Computing, Paul (apart from you!) because we had a full colour advert on the back of the card which accompanied issue 1 and we didn't receive one single response! After much complaining we actually got our cheque for £250 back, unheard of in advertising circles. In fact it was so unusual for anyone to refund advertising money that I kept the cheque as a souvenir. It is dated 2nd November 1984 and is kept in a little tin along with another cheque that someone once sent us for postage, made out for just 22 pence! I didn't bother to cash it!

WE NEED YOUR LETTERS ARTICLES PROGRAMS URGENTLY

PLEASE* CONTRIBUTE *NOW

(It's your mag after all)

Another decent load of letters this issue for which many thanks. Keep your letters coming, folks, they really do mean a lot. As always the address to write to is:

MAILBAG
NEW ATARI USER
P.O. BOX 54
STAFFORD
ST16 1DR



PROGRAMMING

BASIC LISTING SEARCHER

John Foskett shares one of the utilities that help him write so many great programs

When writing or modifying a BASIC program, it is sometimes a necessity to find the number of occurrences of certain things within the listing. As the listing grows, however, the effort needed to search the listing also grows and it can become very time consuming, error prone and very tedious. Sometimes a line number is unwittingly typed wrongly, such as entering line 1015 instead of 1105 and as a result, the line suddenly vanishes into thin air! It is essential that such errors are corrected immediately whilst they are fresh in the programmers mind otherwise they could become forgotten and maybe even result in bugs. It is from this need that the BASIC LISTING SEARCHER was born.

The BASIC LISTING SEARCHER is a utility program that will find every occurrence of a specified string of characters no matter where they are to be found within the listing.

USING THE PROGRAM

Firstly, the input file - the BASIC listing to be searched - must be saved to disk as an ASCII file, that is in the LIST format.

When the BASIC LISTING SEARCHER is run, a lined screen is presented with a prompt to enter the file name of the input file. The default file name of PROGRAM.LST can be entered by simply pressing RETURN and it is convenient to LIST the input file to disk using the default file name.

The next step is to enter the string of characters at the next prompt using a maximum of 12 characters. After entering the string of characters for the search, either RETURN is pressed to continue or ESCAPE is pressed to exit. Upon pressing RETURN the input file will be read from the disk and all lines which contain a match with the character string will be printed on screen.

All lines which contain at least one match with the character string will be printed on screen with all the inverse characters they may contain being converted to normal. All matches with the character string are highlighted in the lines in upper case inverse video for clarity. Note that the input file is not changed in any way by the program, the lines are only printed on screen in this way for clarity.

The search may be paused at any time by

pressing the SPACE-BAR after which RETURN may be pressed to continue or ESCAPE may be pressed to exit.

At the end of the search, the number of times the character string appeared in the listing is displayed on screen together with the number of lines that the character string appeared on. At this point ESCAPE may be pressed to exit or RETURN may be pressed to list the line numbers upon which the character string appeared after which ESCAPE must be pressed to exit.

TECHNICAL DETAILS

ERROR TRAP

The error trap routine is responsible for detecting the End Of File (EOF) error 136 and returning control back to the main program. The error trap routine is limited to only the errors associated with reading a disk file.

DISPLAY LISTS

The program uses two display lists, a normal mode zero, but lined, screen is used for the main display and a special four-line mode zero screen is used for displaying disk errors. The display lists are defined together as ST$ and MOVEd into page 6 at address 1536. The address of the main screen display list is 1536 and the address of the error trap display list is 1590.

THE CURSOR

The program uses player zero as the cursor for keyboard entry and the VBI routine is used to flash its colour between the two preset values (144 and 150). ST$ loaded with zeroes (the heart character) is used to clear the player stripe by MOVEing 256 bytes of its length into it prior to defining the cursor shape again using MOVE.

PRINTING A LINE ON SCREEN

Prior to printing a line from the input file on screen, the display flag at location 766 is set (POKEd with 1) to enable the ESCAPE-CONTROL characters to be printed on screen without the computer acting upon them. After printing the line, location 766 is reset to zero.

FINDING A MATCH

Once a line from the input file has been read into LN$, the program uses UINSTR (universal INSTR) to find all matches with the character string. The first UINSTR command found on line 240 attempts to find the first match with the character string and therefore determines whether or not a line is to be printed on screen. Once the first UINSTR command has found a match, the second UINSTR command on line 320 attempts to find more matches with the character string.

Each time a match is found, the line number is stored in ST$ for listing if required after the search. The line numbers are stored using the normal 2 byte lo/hi method.

VBI ROUTINE

A small deferred VBI routine defined as a string (VBI$) is used to disable all lower case and inverse characters to ease keyboard entry. Because UINSTR is used to find all matches, the status of any letters in the character string is unimportant since UINSTR will find all matches irrespective of case or mode. The routine is also responsible for flashing the colour of the PMG cursor between two preset values (144 and 150), for disabling the attract mode and for disabling the CONTROL-1 stop-start toggle.

PROGRAM BREAKDOWN

To help study the listing, a breakdown of the programs procedures, labels, strings and major variables follow....

PROCEDURES

BEEP	The beep
CLICK	The keyboard click
CURSOR	Controls the horizontal position of the cursor. Used in the INPUT procedure
INIT	Initialising routine
INPUT	For entering data from the keyboard
IPSUB	For printing data on screen. Used in the INPUT procedure
SPACES	Removes any leading and lagging spaces from I$
WAIT	Pauses the search

THE ONE LABEL NAME

# RERUN	Resets the VBI vector and reruns the program

STRINGS

CH$	The character string entered from the keyboard converted into inverse characters
CH1$	The character string copied from CH$ before converting into inverse. As entered from the keyboard
CUR$	For defining the cursor
F$	The file name copied from FN$ but with "D"
FN$	The file name entered from the keyboard without "D"
I$	Stores data entered from the keyboard used in the INPUT procedure
L$	Defined as a line
LN$	Stores the lines of BASIC read from the input file
N$	Line number of the current line as a string
ST$	Used for loading the display lists into page 6, for clearing the PMG stripe and for storing the line numbers of the lines which match the character string
T$	The programs title
VBI$	The VBI routine

MAJOR VARIABLES

CUR, MAX, POS, PRV, XX and YY - General variables used in the INPUT procedure

FND(From UINSTR) - Equals zero if no match is found and exits the loop. If greater than zero, a first match has been found

LCH - Length of CH$

LINES - Number of lines that a match has been found on

LNUM - Line number of the current line

MATCH - Number of matches found which includes more than one per line

PMB - Address of PMG

ST - Pointer for the next position in ST$ to store the next line number upon which a match has been found

THE LISTING

The full listing can be found on this issue's disk. If you prefer to type in the listing a TYPO coded printed listing is available on request, see inside back cover for details.

HEY! HEY! it's The TIPSTER

WORM IN PARADISE

As promised last issue we continue James Matthrick's exploration of Silicon Dreams with this expose on Worm In Paradise. Hope it helps you out!

Firstly, the hint way back in Issue 71 is incorrect - sorry, Steve Nicklin! Getting into the Main City is easy - there is a time limit on the simulation game and so, sooner or later, you will end up in the Main City, but the simulation is a good opportunity to get points.

Solution to the simulation

Get bench, S, E, drop bench, get on bench, get apple, stand up, eat apple, W, W, W, N, get on Behemoth, E, S, W, W, get scale, W, N, N, N, N ... and that leaves you with 80 points.

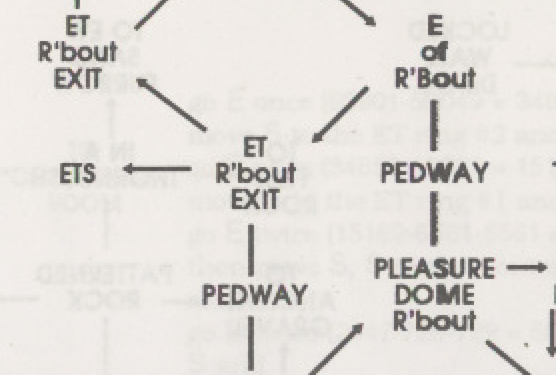
THEN **drop visor, S, E, wear visor** to gain information and a further 40 points. Then you can leave the Dream Palace.

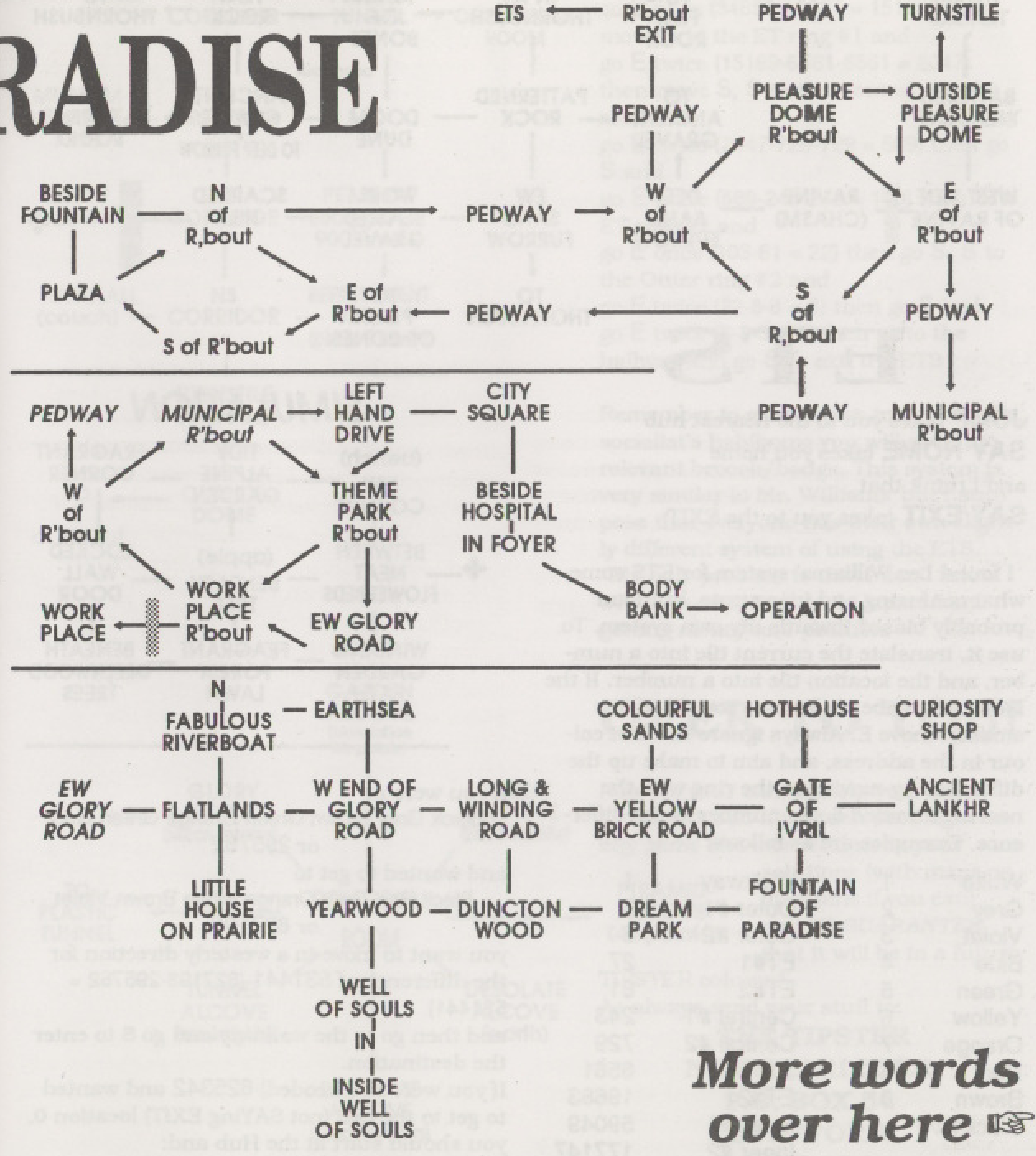
** BIG MAJOR GLITCH **

If you go to the police station and GIVE ME to the officer, you will find yourself in Nevernever land, along with every object and cast member. Double whoops, Level 9! You cannot get out of this bug, but should you EXAM ALL, you will receive a name and description of everything. You can also talk to people, and try objects (e.g. OPEN BOTTLE, PUSH MIRROR) but if you wake the Behemoth, remember there's nowhere to run to.

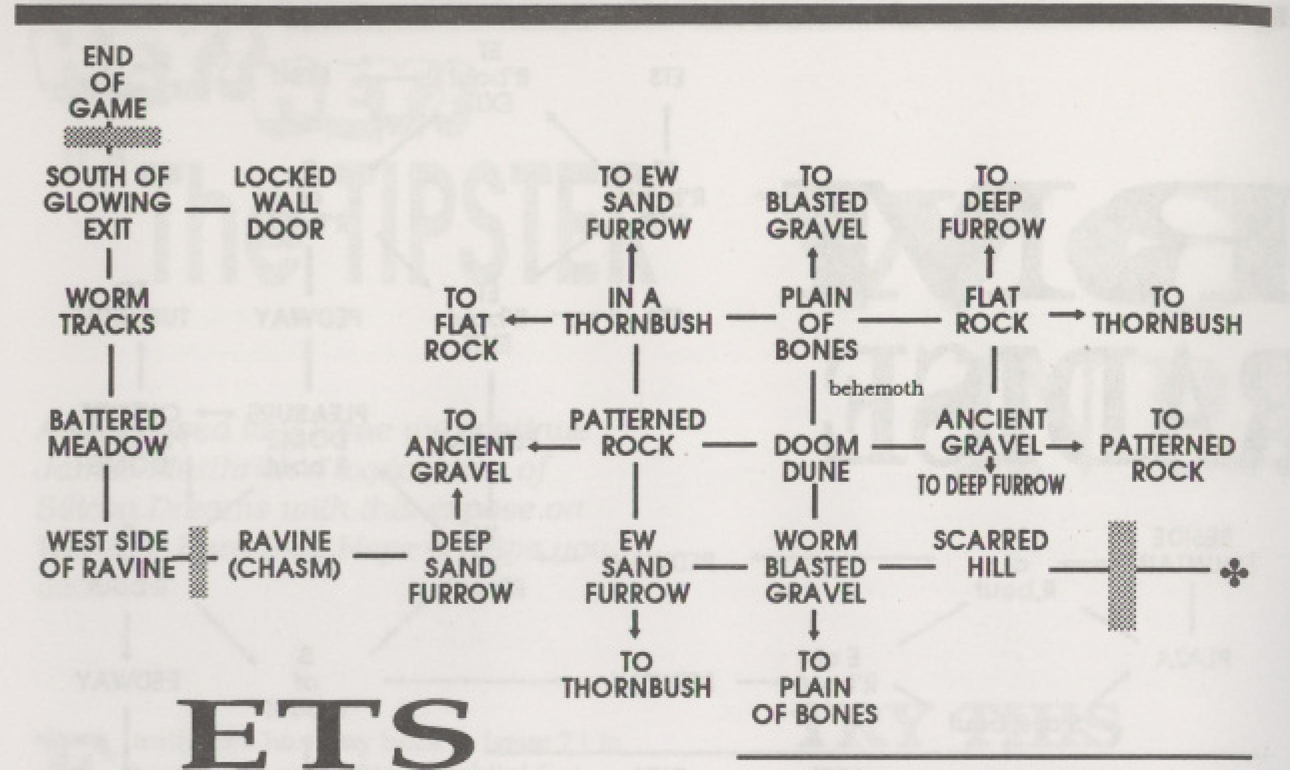
TRY THIS

Should you get caught by a fuzbot, and get fined, you can OOPS as in Return to Eden to renew your creds. It may be worthwhile cashing in your assets at the bodybank if you're making money. Once you've noted the contents of the socialist's wallet, return both to the police station for a reward. Going to work will also earn you money, but you will have to go through a day's 'training' at the job centre - a useful tip for beating the curfew (you can also use the habihall for this purpose. I believe that the curfew is an incentive for the player to find his habihome). Should you take manual work and plug yourself into the walbot, you can get the valve free from the warehouse, but you can still get arrested, even if your walbot is breaking curfew. Does anybody know any other use for the walbot? If you take office work, how do you get down the trapdoor? What use are the vidcam, dagget/batpack, pizza, box, pie, plug, plate, flowers and newspaper. Are they all red herrings? And does anyone know how to get the ticket from the travel agent? I've heard of an alien's seat of Power, but I cannot find it, or the invitation to the party. Having gone to the socialist's home, I cannot find a use for that either, except that it makes me smell pretty bad, until I take a shower.





More words over here ☞



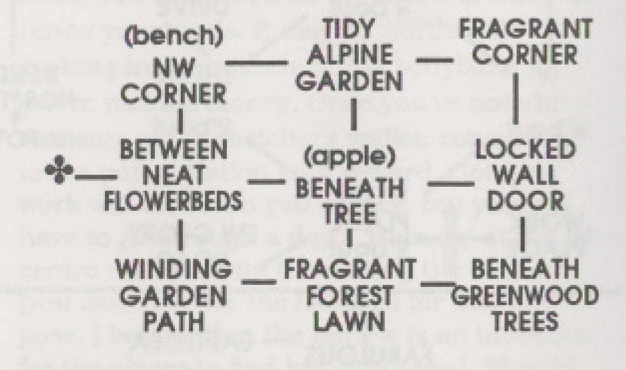
ETS

JUMP takes you to the nearest hub
SAY HOME takes you home
and I think that
SAY EXIT takes you to the EXIT.

I found Les Williams' system for ETS somewhat confusing and inaccurate, but I am probably biased towards my own system. To use it, translate the current tile into a number, and the location tile into a number. If the location number is larger, move W, if it is smaller. move E. Always ignore the first colour in the address, and aim to make up the difference by moving in the ring with the nearest rounded down number to the difference. Examples are as follows:

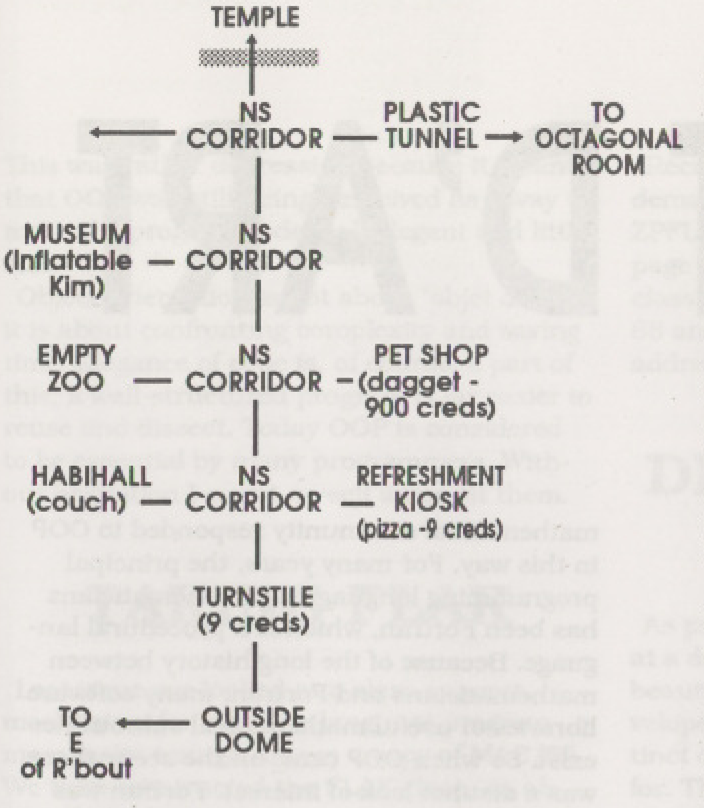
White	1	Walkway	1
Grey	2	Outer #1	3
Violet	3	Outer #2	8
Blue	4	ET#1	27
Green	5	ET#2	81
Yellow	6	Central #1	243
Orange	7	Central #2	729
Red	8	ET#1	6561
Brown	9	ET#2	19683
Black10/0		Inner #1	59049
		Inner #2	177147
		By Hub	531441

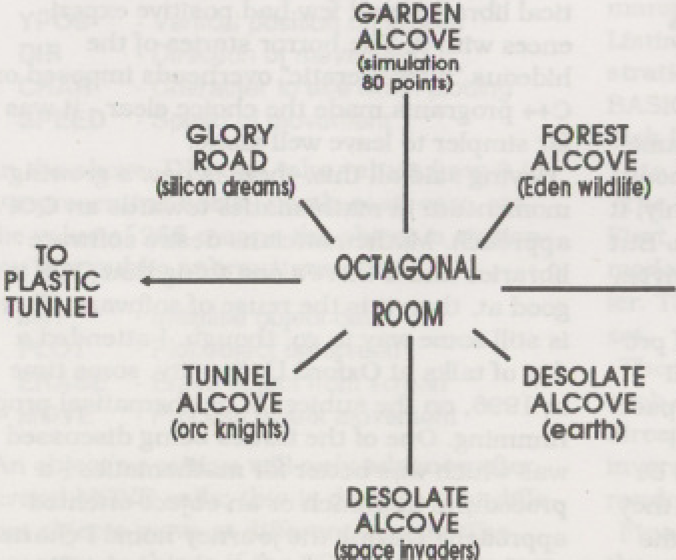
SIMULATION



If you were at
Black Grey Brown Green Orange Green Grey
or 295752
and wanted to get to
Black Red Grey Orange White Brown Violet
or 827193
you want to move in a westerly direction for the difference of 531441 (827193-295752 = 531441)
and then go to the walkway and go S to enter the destination.
If you were at (decoded) 625342 and wanted to get to the exit (not SAYing EXIT) location 0, you should start at the Hub and:
go E once (625342-531441 = 93901) and move S, S to the Inner ring #1 and

PLEASURE DOME





go E once (93901-59049 = 34852) then move S to the ET ring #2 and
go E once (34852-19383 = 15169) then move S to the ET ring #1 and
go E twice (15169-6561-6561 = 2047) then move S, S to the Central ring #2 and
go E twice (2047-729-729 = 589) then go S and
go E twice (589-243-243 = 103) then S to ET ring #2 and
go E once (103-81 = 22) then go S, S to the Outer ring #2 and
go E twice (22-8-8 = 6) then go S and
go E twice (6-3-3 = 0) then go to the hallway and go S to exit the ETS

Remember to gain access to your or the socialist's habihome you will need the relevant brooch/badge. This system is very similar to Mr. Williams' but I suppose that everyone has their own slightly different system of using the ETS.
My only problem is that I don't know what I'm supposed to do, apart from getting money and position.

OVER TO YOU!

Now it's your turn to help your fellow gamesters out. If you can shed light on any game like this send us in your solutions (with maps or diagrams if you can) and I'll ***GUARANTEE*** that it will be in a future TIPSTER column!
As always send your stuff to:

THE TIPSTER
NEW ATARI USER
P.O. BOX 54
STAFFORD
ST16 1DR



OBJET D'ART

Joel Goodwin concludes his series for the more advanced programmer

3. Coup de Grace

Object-Oriented Programming (OOP) became the New Big Thing in the mid 80s. Everybody had to have a piece of the action - suddenly, it was the obvious answer to every problem. But as with any new phenomenon driven by hype, it became misunderstood which in turn meant that it became misused. Stories of projects becoming unnecessarily over-complicated and programs running at a snail's pace started to circulate. The trouble with OOP was that it was not something that could be learnt quickly. People tried to run before they could walk and stumbled; some blamed the new programming "fad" as no more than some computer scientists' idea of a tidy program.

My own experience verifies this as the mathematical community responded to OOP in this way. For many years, the principal programming language for mathematicians has been Fortran, which is a procedural language. Because of the long history between mathematicians and Fortran, many software libraries of useful mathematical subroutines exist. So when OOP came on the scene there was a distinct lack of interest. Fortran was tried and tested, comfortable and the subject of recent revisions. What more could you want? The language that was being pushed by industry, C++, lacked extensive mathematical libraries and few had positive experiences with it. The horror stories of the hideous "bureaucratic" overheads imposed on C++ programs made the choice clear - it was far simpler to leave well alone.

Having said all this, there is now a growing momentum in mathematics towards an OOP approach. Mathematicians desire software libraries and if there's one thing that OOP is good at, then it is the reuse of software. There is still some way to go, though. I attended a day of talks at Oxford University, some time in 1996, on the subject of mathematical programming. One of the issues being discussed was which was better for mathematics - a procedural approach or an object-oriented approach? During the journey home I chatted to a lecturer about the Fortran versus C++ debate. He said it was about which was best - truth or beauty? And truth, he added, won.

This was rather depressing because it meant that OOP was still being perceived as a way to make the program code look elegant and little else.

Object-orientation is not about "objet d'art", it is about confronting complexity and saving time. Elegance of code is, of course, a part of this; a well-structured program is far easier to reuse and dissect. Today OOP is considered to be essential by many programmers. Without hesitation I count myself amongst them.

TAKING FLAK

Last issue, we looked at a class support mechanism for machine language programmers lucky enough to own a copy of MAC/65. We then constructed the FLAK class, an object of which represents a moving graphics mode 0 character. An object holds the following pieces of public data:

XPOS	- Horizontal position on screen
YPOS	- Vertical position on screen
DIR	- Direction of movement
CHAR	- Character to use when plotting
SPEED	- Speed of movement

In the above, DIR can take values from 0 to 7, representing north, north-east, east, etc. The value of 255 means the object is stationary. The public subroutines are:

INIT	- Initialise object data
PLOT	- Plot object on screen
ERASE	- Erase object from screen
MOVE	- Process object movement

An object's position will only advance after several MOVE calls; this is done so that different objects move at different speeds. The exception to this is if the SPEED is set to zero where the object's position will be altered on every MOVE call.

Recall that the FLAK class file made two demands on the main program; the label ZPFLAK must be given the address of a zero page vector which can be used by the FLAK class subroutines, and the zero page locations 88 and 89 must hold the screen memory address.

DEMONSTRATION TIME

As promised last issue, we are going to look at a demonstration of the FLAK class. The beauty of object-orientation is that code developed for a class is valid for as many distinct objects of that class as we have memory for. The whole point of classes is to isolate structure from a particular implementation, to enable reuse of that structure. So to demonstrate the distinct reduction of complexity that OOP achieves, the demo program will manage 120 different FLAK objects at once. Listing 1 is the MAC/65 listing of the demonstration. Listing 2 (on this issue's disk) is a BASIC listing which will create an executable disk file of the demo program (use DOS option 'L' to load FLAKDEMO.OBJ).

The MAC/65 code is not difficult to follow. First, the program creates a blank graphics mode 0 display by using S:, the screen handler. The cursor is erased and the colours are set.

Then the program deals with initialising each object. Each object is given a random direction and speed and is represented by an inverse character. Each object is also given a random position to start from.

Finally, the program deals with processing the movement of each object, which just involves JSR ERASE, JSR MOVE and JSR PLOT. There is some baggage associated with

```
        .OPT NO LIST
;
; Demonstration of FLAK class
; by Joel Goodwin 11-1-97
;
        *= $5000
OBJECT =   $CB
        .INCLUDE #D:OBJECTS.M65
ZPFLAK =   $CD
        .INCLUDE #D:FLAK.M65
;
; Labels
;
RTCLOK =  20          ;Clock lo byte
ICCOM  =  834
ICBAL  =  836
ICAX1  =  842
RND    =  53770       ;random no.
CIOV   =  $E456
;
; Program parameters
;
TOTAL  =  120         ;no. of FLAKs
TEMPO  =  254         ;Speed of prog.
;
; Block of FLAK objects
;
BLOCK     CLASS       FLAK,TOTAL
;
; Miscellaneous data
;
NUM       BYTE        ;Block index
SHAND     .BYTE "S"   ;Used for IOCB 6
;
; Main program
;
; * Initialise display
START  LDX  #$60
       LDA  #12
       STA  ICCOM,X
       JSR  CIOV      ;CLOSE #6
       LDX  #$60
       LDA  #3
       STA  ICCOM,X
       LDA  # <SHAND
       STA  ICBAL,X
       LDA  # >SHAND
       STA  ICBAL+1,X
       LDA  #0
       STA  ICAX1,X
       JSR  CIOV      ;GRAPHICS 0
       LDA  #2
       STA  710
       LDA  #48
       STA  712       ;Colours
       LDY  #2
       LDA  #0
       STA  (88),Y    ;Erase cursor
; * Initialise block
       LDA  #TEMPO
       STA  RTCLOK
       JSR  FIRSTFLAK
ST0    JSR  INIT
       LDA  NUM
       AND  #$3F      ;This makes
       ORA  #$80      ;127<CHAR<192
       OPUT CHAR
       LDA  RND
       OPUT SPEED     ;SPEED random
       LDA  RND
       AND  #$07
       OPUT DIR       ;DIR random 0-7
ST1    LDA  RND
       AND  #$3F
       CMP  #40
       BCS  ST1
       OPUT XPOS      ;XPOS rand. 0-39
ST2    LDA  RND
       AND  #$1F
       CMP  #24
       BCS  ST2
       OPUT YPOS      ;YPOS rand. 0-23
       JSR  PLOT
       JSR  NEXTFLAK  ;next FLAK obj.
       BNE  ST0
; * Wait for VB before moving
WAIT   LDA  RTCLOK    ;Is 1 VB over?
       BNE  WAIT
       STA  77        ;Kill attract
       LDA  #TEMPO    ;Start timer for
       STA  RTCLOK    ;next step
; * Process block objects
       JSR  FIRSTFLAK
PROCESS JSR ERASE
       JSR  MOVE
       JSR  PLOT
       JSR  NEXTFLAK
       BNE  PROCESS
       BEQ  WAIT      ;finished block
;
; FIRSTFLAK subroutine
; Set OBJECT to start of block
;
FIRSTFLAK LDA # <BLOCK
       STA  OBJECT
       LDA  # >BLOCK
       STA  OBJECT+1
       LDA  #TOTAL
       STA  NUM
       RTS
;
; NEXTFLAK subroutine
; Move to next object in block
;
NEXTFLAK CLC
       LDA  OBJECT
       ADC  #FLAK
       STA  OBJECT
       LDA  OBJECT+1
       ADC  #0
       STA  OBJECT+1
       DEC  NUM
       RTS
;
; RUN address
;
       *=   $02E0
       .WORD START
```


running through a block of FLAK objects (see the FIRSTFLAK and NEXTFLAK subroutines), but this is superfluous to the core of the demo which is the FLAK class itself.

The program monitors the timing of the movement loop carefully. It is possible that one cycle of movement will be quicker than the next, because many objects may not have physically moved at all. So after the FLAK object movements have finished, the program will wait to make sure each cycle takes exactly the same time. It uses the zero page timer RTCLOK (locations 18, 19, 20) to make sure the duration of each movement cycle is the same. Another reason for waiting after each cycle is that it is possible that certain cycles may be so quick that several cycles will have been processed before the display has had time to show one of them. The 6502 processor can get through plenty of machine language before the end of a single TV frame; by forcing the length of a cycle to be at least one frame, each cycle is always visible.

There are two parameters which can be altered to affect the program behaviour. The first parameter, TEMPO, governs the duration of the movement cycles. It can vary from 0 to 255. The program, as given in listing 1, does not use the fastest TEMPO value because the work involved in some cycles exceeds that allowed by TEMPO=255; the result of using a excessive TEMPO value is frequent pausing in the motion. The second parameter, TOTAL, is the number of FLAK objects to be managed. The program is designed to handle up to 256, but it could be easily modified to cope with more than this. Of course, the higher TOTAL becomes, the lower TEMPO must be.

Consider how you might have handled writing a program designed to throw 120 characters around the screen, each in different directions with different speeds. It is common, in such a situation, to define arrays for each piece of data. The difference here was that we arranged the data into groups which belonged to the same FLAK object rather than, for example, separate speed and position arrays. Although the array approach has optimisation aspects to vouch for it, the class approach is quicker, simpler and easier to debug (trust me on this last one). Please bear in mind, however, that the class could have been made faster, without breaking an OOP approach, by some restructuring of the subroutines. This has not been done as it would have made the class more complicated than was necessary for the purpose of these articles.

The OOP approach is not just better because of the development speed, though. We could take the FLAK class and apply it in a different program if we wanted, with no hassle at all. But what do we do if we wanted to extend the class?

INHERITANCE IN MAC/65

Primitive inheritance can be achieved in MAC/65 with little effort. To derive a new class DERIVED from another class ORIGINAL then we need to use a special instruction after the NEWCLASS declaration. The first line of our class definition must read "CLASS ORIGINAL":

```
         NEWCLASS
         CLASS ORIGINAL   ;Inherit from ORIGINAL
DATA1    BYTE
DATA2    DBYTE
         ....
DERIVED  ENDCLASS
```

Recall what inheritance means. The DE-

RIVED class will contain all the data that was in ORIGINAL and all of ORIGINAL's subroutines will work on DERIVED!

A specific example might be more instructive. Consider the FLAK class created last issue. We want to use the class for a game, where each FLAK object is a target which is worth some points when hit. We decide that it would be better to add the public data member SCORE to the FLAK class to facilitate this. Conventional programming would have us go back to the FLAK class file and "hack" in the SCORE member. OOP suggests we inherit a new class from the old using inheritance, as follows.

```
         NEWCLASS
         CLASS FLAK   ;Inherit from FLAK
SCORE    BYTE         ;New SCORE member
GAMEFLAK ENDCLASS
```

Our game would then use the GAMEFLAK class throughout. We would need to .INCLUDE the appropriate files, of course - both the FLAK and GAMEFLAK class files are required and must be .INCLUDEd in that order. This example may appear trivial but this is only scratching the surface. Inheritance can permit many more additions than just a SCORE. The derived class can also have its own subroutines, public and private.

Now would be the time to point out limitations in this approach. MAC/65 cannot support "multiple" inheritance - the ability to derive a class from two different classes simultaneously. The following would NOT work:

```
      NEWCLASS
      CLASS MAN     ;Inherits from MAN
      CLASS WOMAN   ;Does NOT inherit from
                    WOMAN
CHILD ENDCLASS
```

What you could do to bypass this problem is inherit from one class and then nest an object from the other class as a data member, i.e.

```
       NEWCLASS
       CLASS MAN     ;Inherit from MAN
MOTHER CLASS WOMAN   ;MOTHER data member
CHILD  ENDCLASS
```

The CHILD class only truly inherits from one class. The members of MOTHER need to be accessed indirectly. Supposing JOHN is a CHILD object, then references to the encapsulated MOTHER object must be made in a JOHN+MOTHER+<woman member> style. Similarly, WOMAN subroutines cannot be used on JOHN, they must be used specifically on JOHN's MOTHER object. This appears to disturb the symmetry in the CHILD example and it would probably be better not to inherit from the MAN class after all; nesting it as an object called FATHER makes more sense. The idea of nesting objects within objects is not necessarily bad. If we go back to our trustworthy POSITION class then we could define a POLYLINE class containing several POSITION objects from which a line can be described. To inherit POLYLINE from POSITION makes no sense.

Another drawback in MAC/65 inheritance concerns the private members of the original class. When deriving a new class, the new class unfortunately cannot access any of the private members from the original class. The .LOCAL shield around the original class file renders the private data and subroutines invisible to the code in the derived class file. There is no way of negotiating this satisfactorily.

You may be interested in learning how inheritance works in MAC/65. Consider nesting an object of one class as the first member in the definition of a new class. The data members of the old class are indexed by their labels, e.g. the first member corresponds to the first byte, the second member, perhaps, to the third byte and so on. Because this class is nested right at the start of the new class then all these indices remain EXACTLY THE SAME for the new class. That is, the first member of the old class corresponds to the first byte in the new class, the second member to the third byte and so on again. Therefore, we need not treat these data members as if belonging to the old class; we can pretend that they were part of the new class all along. The inheritance falls out naturally.

POLYMORPHISM IN MAC/65?

Polymorphism is a lovely idea but can we mimic it using MAC/65? No. Polymorphism, for our purposes, means that classes derived from a common class can inherit a subroutine call, but replace the actual subroutine.

Suppose we derived two different classes from the FLAK class: DEADFLAK and LIVEFLAK. The DEADFLAK objects do not move and the LIVEFLAK objects move changing their direction frequently. Polymorphism would allow us to implement new MOVE subroutines for both DEADFLAK and LIVEFLAK which would mean the main program would never have to determine whether it was dealing with a DEADFLAK or LIVEFLAK object. All the program would need would be JSR MOVE and the correct subroutine would be called automatically. No, we cannot do this. This is not so bad because polymorphism, in some OOP languages, must be pre-planned. That is, we should have declared the FLAK class subroutine MOVE as a candidate for polymorphism from the outset.

What we can do is create a data member which holds a subroutine address. Instead of a set MOVE subroutine, we have a MOVE data member:

```
     NEWCLASS
.....
MOVE DBYTE
FLAK ENDCLASS
```

The main program would not use JSR MOVE; it would have to pull out the two MOVE bytes and create the correct JSR instruction while running. A macro could be created to do this, call it PJSR, so the main program could simply state PJSR MOVE.

Note that the MOVE subroutine can be different for two objects of the same class. If this was true polymorphism, then MOVE would only be different for objects of different classes. But then again, you can't have everything.

That now wraps up everything I wanted to say about using classes in MAC/65 projects. So where do we go from here? What do we do with this new potential?

POTENTIAL

Many program aspects lend themselves very well to OOP methods. When I wrote Motivation (NAU issue 78), I had object-orientation in mind even though OBJECTS.M65 hadn't been developed then; player-missile graphics are ideal for applying OOP ideas to. Look at the structure that the Basic programmer sees. Each player has several inputs: IMAGEADR, HPOS, VPOS, SIZE and COLOUR. Many of the untidy aspects of PMG are removed and encapsulated in the VBI. It even destroys the identity of each missile, coalescing them into an independent player almost seamlessly. Private data belonging to each

player is hidden from the user; Motivation secretly records the vertical position and image length used by the last VBI. This is what OOP is all about. We cannot see any of the machinery and we never need to. Motivation, of course, is not perfect as I stated in the accompanying article. There is plenty of scope for improvement, to strengthen the object-oriented aspects.

A project which cries out for an OOP approach is a windows graphical-interface system. Each window is perceived as independent entity; why not develop a windows program in the same way by using OOP? Think of what data and subroutines should be attached to a WINDOW object. Are there different types of windows? Would inheritance play a part? How can we involve a "pointer" to interface with a window? And so on.

One problem I've faced when doing extensive machine language work is that when graphical changes are issued they can sometimes end up out of sync. Suppose I wanted to change the entire screen display; all of the player-missile graphics must change, a different display list will be selected and new display-list interrupts will be invoked. If I simply programmed up these changes one after another then it is likely that there will be a brief flash of chaos as some of these changes become visible before all of the changes have been finished. The easy way out is to turn ANTIC off with the POKE 559,0 trick, do all the changes, then turn it back on. This works but cannot give the impression of shifting from one situation to another continuously. The brief black screen is a pause in flow. I've always wanted to be able to change the entire screen display without such an interruption. All of the changes can be delegated to the VBI to accomplish this, but it gets very complicated, very quickly. OOP helps because by constructing a SCREEN class consisting of all the important screen data, the information is centralised and the situation is controlled.

We have examined just three situations where OOP can help. There are many, many more. Some of these are not obvious but all are effective in reducing complexity and improving reusability.

CODA

It has been a long haul, but it is finally time to conclude our discussion of OOP. The aim of Objet D'Art was to confer some new programming ideas to the reader and to demonstrate their benefits. This is not easy to do in just three articles. The principles of object-orientation have been explored far more thoroughly elsewhere; what you have read here is merely a gross simplification of OOP. There are plenty of books on the subject, but sadly I have found many of them inaccessible to the casual programmer. Most are aimed at the those fluent in modern programming terminology and normally experienced in the language C.

Nevertheless, I hope I have stimulated your interest in something different. We have to remain aware of what is going on around us as the world moves on and there is always more to learn. Just as object-orientation has been established as the be-all and end-all, its flaws are already being examined. OOP is not the "ultimate" approach, it is just the latest one. New buzzwords are invented every day - for example, have you ever heard of polytypic programming? Who knows what the next revolution is or what it will bring? We must all keep our eyes open and be prepared to accept new ideas - otherwise we will quickly become nothing more than a collection of interesting antiques. ●



HAPPY TYPER

by Steve Hillen

Dennis Fogerty revives one of his favourite programs from Monitor magazine

Long ago when I did a lot of programming and typing of listings on my XL I found this program a godsend. Today, hopefully, others will find it helpful too. All credit must go to Steve Hillen, (where are you now?), and I hope he won't object to my reproducing his article here verbatim.

INTRODUCTION

The Happy Typer is a utility for use with Atari Basic. It will give you automatic intelligent line-numbering and 10 extra keys which you can redefine to print out keywords, thus speeding up your typing. Unlike many auto line numbering facilities, this one allows the full use of the Atari screen editor, so you can adjust lines while still in the auto mode. The redefined keys are accessed by pressing the SHIFT and CONTROL keys simultaneously with a number key. These keys are not used by Basic or the operating system, so you can still type in all those control characters.

MAKING A BOOT DISK

For a disk system, type in the listing and save it. Type RUN and the program will check your typing and ensure that the data is correct. Retype those lines that produce an error. Once it is ready the program will ask you to insert a disk with DOS on it. The program will then save out an Autorun.sys file onto the disk. Don't change the filename - Happy Typer will only work as an Autorun.sys file. The next time you boot this disk with Basic, Happy Typer will be ready for use.

USING HAPPY TYPER

THE AUTO LINE NUMBERING

Every time you tap the TAB key after a RETURN, a new line number will be printed. If you type on the TAB key and the last key pressed was not a RETURN then the TAB will be performed as normal. This is better illustrated by example.

Directly after power-up, press the TAB key - the first line number will be printed. Type ?"hello" then RETURN then TAB. The next line number will appear. Play around and get used to using the TAB key after a RETURN. If you type in a few lines of Basic, then list them, then press RETURN and TAB, the next line number after the last line of your program is printed. Also, if you use the cursor

```
EI 1 REM ********************************
DZ 2 REM *  HAPPY TYPER (DISK VERSION) *
NP 3 REM *       BY STEVE HILLEN        *
EI 4 REM *  MONITOR MAGAZINE. ISSUE 9 *
EM 5 REM ********************************
DW 10 DATA 0,1,2,3,4,5,6,7,8,9,0,0,0,0,0,
   0,0,10,11,12,13,14,15
MM 15 DIM DAT$(91),HEX(22)
GZ 20 FOR X=0 TO 22:READ D:HEX(X)=D:NEXT
   X:LINE=990:RESTORE 1000:TRAP 60:? "Che
   cking data":?
LI 25 LINE=LINE+10:? CHR$(28);"Line:";LIN
   E:READ DAT$:IF LEN(DAT$)<>90 THEN 110
VD 28 DATLINE=PEEK(183)+PEEK(184)*256:IF
   DATLINE<>LINE THEN ? "Line :";LINE;" m
   issing.":END
LU 30 FOR X=1 TO 89 STEP 2:D1=ASC(DAT$(X,
   X))-48:D2=ASC(DAT$(X+1,X+1))-48:BYTE=H
   EX(D1)*16+HEX(D2)
JW 35 IF PASS=2 THEN PUT #1,BYTE:NEXT X:R
   EAD CHKSUM:GOTO 25
BL 40 TOTAL=TOTAL+D1+D2+96:IF TOTAL>999 T
   HEN TOTAL=TOTAL-1000
CE 45 NEXT X:READ CHKSUM:IF TOTAL=CHKSUM
   THEN 25
ME 50 GOTO 110
PE 60 IF PEEK(195)<>6 THEN 110
IL 65 IF PASS=2 THEN PUT #1,224:PUT #1,2:
   PUT #1,225:PUT #1,2:PUT #1,78:PUT #1,3
   4:CLOSE #1:? "Done it.":END
JK 70 ? "Insert disk with DOS. Press <ret
   urn>.";:DIM IN$(1):INPUT IN$:OPEN #1,8
   ,0,"D:AUTORUN.SYS"
HH 90 PUT #1,255:PUT #1,255:PUT #1,0:PUT
   #1,31:PUT #1,176:PUT #1,34
CN 100 ? "Writing file":? :? :PASS=2:LINE
   =990:RESTORE 1000:TRAP 60:GOTO 25
AZ 110 ? "Bad data on line:";LINE:LIST LI
   NE:? TOTAL:END
FC 1000 DATA 20FFFFD8A94E8DE702A9228DE802
   A200BD1A03E8E8E8C945D0F6BD180385CBBD19
   0385CCA91B9D1803A9229D19,243
HV 1010 DATA 03A00FB1CB991B228810F8AD1F22
   1869018D5020AD202269008D5120A9208D2022
   A94E8D1F22AD212218690 18D,232
XY 1020 DATA 0D21AD222269008D0E2178AD0802
   8DA51FAD09028DA61FA91F8D0902A97E8D0802
   58186078D88A489848AC1822,310
AK 1030 DATA AD09D28D1822A209DD4422F013CA
   10F8C92CD004C00CF02B68A868AA584CFFFFBD
   6421F01B8D63218A0A0A0A0A,501
XP 1040 DATA AA8E6221BD6E21200C21AE6221E8
   CE6321D0EE4C9F1FA2FFE8BD0E22C930F0F88E
   1922200C21EE1922AE1922BD,707
EP 1050 DATA 0E22E00590F0A9218DFC0268A868
   AA685840A2FFE83056BD8005C920F0F6C930F0
   F29049C93AB045E8A000990E,823
XK 1060 DATA 22BD8005297FC930900CC93AB008
   E8C8C00590EAF014A204B90E229D0E22CA8810
   F6A9309D0E22CA10FA18A204,926
CD 1070 DATA BD13227D0E22C93A9003E90A389D
   0E22CA10ED6020FFFF08C99BD0388A48984878
   20F31F58A2008E4322A000B9,31
GS 1080 DATA 8005DD2B22D008E8C8C00390F2B0
   1DC8E8C003D0FAEE4322E00590E1A9FF8D1A22
   68A868AAA99BEE1A222860AC,229
RO 1090 DATA 1A22A99B9980058D8005AE4322BD
   33228DD620BD31228DD72020D520A2008E1A22
   901AA9099D42038E4903A97F,251
SK 1100 DATA 9D4803A9229D4503A9359D440320
   56E44C4DA04CFFFFA0FFC8B98405C99BF00BC9
   309004C93A90F038B01E8830,393
AB 1110 DATA FAA204B98405290F9D1322CA8810
   F4E0003008A9009D1322CA10FA186020FFFF60
   A004B98005C9309046C93AB0,427
IB 1120 DATA 4238E930480A0A0A0AAAA9008D61
   21C8B98005C99BD00968AAA9009D6421F021C8
   B98005C99BF0119D6E21E8EE,543
DU 1130 DATA 6121AD6121C91090EACE612168AA
   AD61219D642118603860000000000000000000
   000000000000000000000000,215
```

Underline = INVERSE CHARACTERS · [ ] = CONTROL + CHARACTER · < > = INVERSE CONTROL + CHARACTER

keys to move to another line, and press RETURN on that, then the next auto line will be that plus 10.

This method of auto line numbering remembers the last line you typed, and gives you the next one if selected with the TAB key.

Finally, to change the increment of the auto line numbering, just type: INC nn where nn is any number you like. e.g. INC 2000 will give line numbers 2000 apart, INC 1 will give numbers one apart.

THE REDEFINED KEYS

The keys that can be redefined are the row of 10 numbers across the top of the keyboard. To redefine a key type:DEF 1 ?"hello".

Every time you type SHIFT CONTROL 1 simultaneously, ?"hello" will be printed. Another example: DEF 8 POKE. Now a shift-- control 8 will type out POKE. Get the idea?

Note that the space between DEF 1, and the string are important, also the space between INC and nn. Also note that each key is allocated 16 characters, so this is the maximum you can stuff into one key. If you redefine a key that has already been redefined, then the last redefinition will be printed. If you should wish to delete a key just type DEF 1 then RETURN without the second space. You will find that all characters, except trailing spaces and the return key, can be printed, so you might set up one key to backspace, say, 10 characters by using the Escape Cntl cursor keys.

Finally, don't worry about hitting the System Reset key, the program is safely installed and protected, and will remember all the keys you've redefined, and the increment and current line number.

```
TM 1140 DATA 0000000000000000000000000000
   00000000000000000000000000000000000000
   000000000000000000000000,535
NH 1150 DATA 0000000000000000000000000000
   00000000000000000000000000000000000000
   000000000000000000000000,855
TW 1160 DATA 0000000000000000000000000000
   00000000000000000000000000000000000000
   000000000000000000000000,175
GO 1170 DATA 0000000000000000000000000000
   00000030303031300000000010000000000000
   000000000000000000000000,512
AF 1180 DATA 00494E434445462021D81053796E
   746178206572726F722E9B00F2DFDEDAD8DDDB
   F3F5F0A90C8D1822A50C8D01,620
JT 1190 DATA 1FA50D8D021FA900850CA91F850D
   A9008D4402A200A9099D4203A9229D4503A98B
   9D4403A9009D4903A97F9D48,684
HR 1200 DATA 032056E44C031F48617070792054
   79706572204F4B2E9B00000000000000000000
   000000000000000000000000,297
```

HOW DOES IT WORK?

The program falls into two sections. Firstly there is an editor patch. The editor is located in the device table, and its vectors are moved into RAM. I adjust the get-byte vector to point to my new routine. This new routine waits for a return to be typed then scans the input buffer for either INC or DEF. If neither are found then the line is passed back to Basic as a normal line. If one is found, then the operation is performed, and the line is not passed to Basic. The second section is a patch into the keypress routine. The keypress interrupt vector is stolen and the new routine looks for a shift-control number or a TAB immediately following a return. If a defined key is detected, then the string is printed out a byte at a time through the editor put-byte routine. If a TAB is found, then the input buffer is examined for a line number and the increment is added to form the next number which is then printed via the put-byte routine. ●



UTILITY

CD COLLECTION DATABASE

by Kevin Cooke

This CD collection database program was written by me in 1997 in an attempt to catalogue my ever growing collection of CDs.

The first question you're probably about to ask is why anyone would actually want to catalogue their CDs! Well, there are a number of reasons for this. Firstly, it gives you a hard record of exactly what you have (and, let's face it, when you've got more than about 50 CD's, you're unlikely to be able to remember them all). This would be useful in the event of an insurance claim, should your collection be stolen or damaged. In my case, I have around 450 CDs, many of which are hard to find or limited edition versions, and are therefore of great monetary and sentimental value to me. [Incidentally, it's well worth keeping receipts and taking photos of your collection regularly in the event of an insurance claim arising).

I also find the program useful because it allows me to search quickly for all of the CDs I have by any one artist, (useful because I sometimes sell CD's). If you wanted to put a number after each CD name, in brackets, you could use the program to tell you where in your collection to find the particular CD you're looking for. This feature is quicker than looking through all CDs by hand, especially as the writing on the spine of a CD case is usually small, in different fonts, and sometimes with the artist and sometimes with the title appearing first.

FIRST GET A WORD PROCESSOR

This relatively short program, written in Turbo BASIC (13 years old and still one of the best versions of BASIC I've ever encountered!), is simply a highly-specialised text file viewer. The four databases are stored as text files (created with a word processor of your choice) and then read by the program. You may criticise my choice of making you use a separate word processor to create your database, but let me explain my reasoning. Firstly, any extra text editor program I wrote would have difficulty competing with the flexibility of a dedicated word processor. Secondly, despite the speed of computers, when you've got a large file of information, nothing can sort things into alphabetical order faster than YOU! It seemed silly to write a program to sort each new entry into it's correct alphabetical database position when it's so easy to do it yourself! Thirdly, I wanted to keep the program as short as possible so that it would load and run quickly - if it didn't do this, it would defeat the whole purpose of the program, which is to save you time.

OPTIONS

When the program loads and runs, you are presented with a menu offering you 7 options. The first two allow you to search for all titles by the artist of your choice in either the album or single database. The next two entries allow you to view all singles or albums in the database in order, in case you can't find the artist that you are looking for in your search using the first two options.

The next two options allow you to look at the database of the music compilations and movie soundtracks stored on the disk.

Finally, the last option quits to DOS.

SAMPLES

On this issue's disk you will find the database ready to run along with four sample database (text) files:

ALBUMS - The album database
SINGLES - The single database
COMPILAT - The Compilation database
SOUNDTRA - The Movie soundtrack database

The format of each database file, as you will see by examining my example files, must be as follows:

For the album or singles database:

SURNAME, FIRST NAME
- 1st Album or single title
- 2nd Album or single title
- 3rd Album or single title
*
NAME OF BAND/GROUP
- 1st Album or single title
*
WET WET WET
- Picture this
- End of part 1
*
END OF FILE

For the movie soundtrack or compilation database files:

- BEVERLY HILLS COP II
*
- BODYGUARD, THE
*
- BOOMERANG
*
END OF FILE

The easiest way to keep to this format is to copy the sample database files, along with the main program, onto a separate disk and edit the sample files to contain the titles of your own CDs.

As can be seen, for the movie soundtrack and compilation databases, you do not have to enter the artist name. This is because the tracks on these CDs are usually by a number of different artists, hence the program automatically displays "Various Artists" under the "Artist" heading when the file is viewed.

TAPES TOO

Of course, I mean no prejudice towards cassette tape users by referring to CDs throughout these instructions. The program is equally as useable for people who only own cassettes, but I am recognising the fact that nowadays a lot of people have CD players and few shops have a wide selection of new albums on cassette tape.

I hope you enjoy using the program. It fulfils my needs, and I hope it fulfils yours. If you have any suggestions or comments, feel free to write to me via Page 6 to let me know.

THE LISTING

This program can be found on this issue's disk ready to run. A TYPO coded type-in listing is available on request - see inside back cover



USEFUL USR

John Foskett explains the easy way to connect machine language with Basic

There is no doubt that USR is a very important instruction in the BASIC language since this is the only instruction with which the BASIC programmer can obtain the advantages of machine code. For instance it is often necessary to move a block of data from one address to another such as when copying the character set into RAM, moving PMGs vertically, etc. Doing this in BASIC can be very time consuming but by using a machine code routine via USR, the process is virtually instant. The advantages of machine code are obvious - much reduced initialising times, smooth PMG movement, animation, etc. and all this is available to the BASIC programmer via USR. Basically, what is awkward and difficult if not impossible in BASIC is a breeze in machine code.

WHAT IS USR?

The USR instruction is used to GOSUB to a machine code subroutine after which control is returned to BASIC in exactly the same way as BASIC gosubs to a BASIC subroutine. In a similar way to passing values into a BASIC subroutine, that is by equating variables prior to calling the subroutine, values or parameters may if necessary by passed into the machine code routine via the USR call. When used without additional parameters, the USR function contains only the address of the machine code routine thus (assuming the routine is at the start of page 6)....

```
X=USR(1536)
```

Alternatively, the machine code routine may be contained within a string as relocatable code (say MC$) in which case USR is used as follows....

```
X=USR(ADR(MC$))
```

In some cases when the machine code routine is relatively small and particularly if only used once, the machine code routine representing MC$ may be placed directly into the USR function as follows....

```
X=USR(ADR("machine code string"))
```

When USR is used with additional parameters, they follow the address of the machine code routine all separated by commas. Assuming the use of TWO additional parameters (A and B) the above three USR functions become....

```
X=USR(1536,A,B)
X=USR(ADR(MC$),A,B)
X=USR(ADR("machine code string"),A,B)
```

THE STACK

When running any program, the computer constantly uses the stack (page 1 of RAM) as a temporary storage for storing addresses and values when actioning subroutines, loops, etc. The stack is a special part of the computers RAM which requires no addressing and no reference of any kind, information is simply stored there, where it is actually put is immaterial. The best way to visualise how the stack works is to imagine a pile of plates where the plates must be put on and taken off the pile in order, thus the last plate on the pile MUST be the first plate to be removed. From this it can be seen that the stack works on the "Last In First Out" (LIFO) principle and so long as this rule is strictly adhered to, there will be no stack errors. This then is the reason why no addressing or referencing of any kind is required in order to use the stack.

WHY MENTION THE STACK?

Normally a programmer doesn't need to consider the stack since its working is fully automatic via the computers operating system. The only time the BASIC programmer must consider the stack is when writing the machine code routine which will be actioned via the USR command. This is because USR stores at least one value on the stack during its operation which MUST be removed prior to returning to BASIC in order to let the computer "see" the return address which it stored there prior to actioning the routine in the first place. If this is not done then it can be seen why the computer may produce stack errors (error 10) or even crash or lock up since it will obviously read the wrong return address prior to returning to BASIC.

THE PARAMETERS

In order to pass any parameters to the machine code routine that the USR command may contain, USR places them on the stack following the return address in a two-byte format. USR stores the low-byte first followed by the high-byte and in the order of the last parameter first which allows their retrieval from the stack to be in a logical sequence which is the first parameter in the USR call first in the order of high-byte first.

THE ODD BYTE

After all the parameters (if any) have been stored on the stack, the USR command stores a further value on the stack, a value of its own, a record of the number of parameters included in the call. This odd byte from now on referred to as the "odd byte" may at first appear absolutely useless, but it can have a very important role to play.

THE MACHINE CODE ROUTINE

As previously stated, all parameters and the odd byte must be removed from the stack by the machine code routine before exiting back

to BASIC and assembly language has the "PLA" instruction which does this. PLA stands for "PuLl Accumulator" which literally pulls the last entry off the stack and loads it into the processors accumulator for actioning by the machine code routine. Note that PLA only removes ONE byte from the stack, therefore two PLAs are required to remove each parameter from the stack even if the value of the parameter is less than 256 in which case the high-byte will be zero. Also note that only ONE PLA is required to remove the odd byte from the stack since it can never exceed 255 if only because of the length of a BASIC line. It therefore follows that unless your machine code routine makes use of the stack itself, there should always be an ODD number of PLAs in the machine code routine.

EXAMPLE ROUTINES

The following example routines are for demonstration purposes only and are designed to change the screen colour to show their working. The following routines assume the machine code routine has been assembled in Page 6 at address 1536. Note that the source code listings and the BASIC programs for the routines have been included on the issue disk for convenience.

ROUTINE 1: A SIMPLE ROUTINE

Source code file name **USR1.SRC**
BASIC demo file name **USR1.BAS**

The following is a very simple routine which changes the screen colour from the normal Atari blue to green. The routine is called using X=USR(1536)

```
PLA
LDA #180
STA 710
RTS
```

The first instruction is PLA which removes the odd byte from the stack and loads its value (zero in this case) into the processors accumulator. This value is unwanted so it is discarded by the next instruction which loads the processors accumulator with the number 180 using LDA (LoaD Accumulator) which is then stored in location 710 using STA (STore Accumulator) where location 710 is the screen colour register. This is the machine code or the assembly language equivalent of POKE 710,180. The last instruction is RTS (ReTurn from Subroutine) which exits the routine and returns control back to BASIC.

ROUTINE 2: USING PARAMETERS

Source code file name **USR2.SRC**
BASIC demo file name **USR2.BAS**

A more flexible version of the above routine (routine 1) which uses a single parameter to pass the screen colour value into the machine code routine. This routine can accommodate any colour value required from BASIC without the need to modify the routine itself. The routine is called using X=USR(1536,A)

```
PLA
PLA
PLA
STA 710
RTS
```

As before the first PLA removes the odd byte from the stack which is then discarded and the next two PLAs remove the one parameter from the stack. Since a colour value cannot exceed 255, only the low-byte is required hence the high-byte is discarded. With the colour value currently stored in the processors accumulator via the last PLA, it is then stored in location 710 and the routine exited via RTS returning control to BASIC as before. Note that if the value of the parameter used in the call is greater than 255, then only its low-byte will be considered.

ROUTINE 3: A CRASHPROOF ROUTINE

Source code file name **USR3.SRC**
BASIC demo file name **USR3.BAS**

To exit correctly from a machine code routine, the correct number of parameters MUST be used in the USR call, thus if a machine code routine is written to use ONE parameter then only ONE parameter must be used within the USR call. The following routine shows how the "odd value" stored on the stack can be used to produce a completely crashproof routine no matter how many parameters are used in the call. The routine will cycle through as many screen colours as there are parameters in the call before returning to BASIC. The routine is called using X=USR(1536,A,B,C,D,......n)

```
     PLA
     BEQ EXIT
     TAX
LOOP
     PLA
     PLA
     STA 710
     LDA #200
     STA 20
DELAY
     LDA 20
     BNE DELAY
     DEX
     BNE LOOP
EXIT
     RTS
```

As before the first PLA removes the odd byte from the stack which is then transferred into the processors X-register using TAX (Transfer Accumulator to X-register). Note that the odd byte is first checked to see if its value is zero using BEQ (Branch if EQual (to zero)) before transferring its value into the X-register. If the value is zero, then there are no parameters in the USR command and the routine is exited via the label EXIT to the RTS instruction to exit back to BASIC. If at least one parameter is used in the USR command, then the loop (identified by the label LOOP) is entered where ONE parameter is removed from the stack via the two PLAs, again only the low-byte is required as before which is then stored in location 710 using STA as before. To allow the routines operation to be seen, a time delay loop has been incorporated using location 20, the low-byte of the computers real time clock. The number 200 is stored in location 20 and the delay loop waits for its value to return to zero before preceding which gives about a 1 second delay. After the delay, the X-register is decremented by 1 using DEX (DEcrement X-register) and checked to see if its value is equal to zero using BNE (Branch if Not Equal (to zero)) and if zero the routine is exited via RTS as before. If the X-register does not contain a zero at this point, the loop is executed again to remove the next parameter from the stack which is then actioned as before. Note that the X-register always contains a record of the number of parameters remaining on the stack during the operation of this routine, it is therefore only a matter of checking the X-register to see if there are any more parameters on the stack and if so, remove them for actioning, but if not, exit back to BASIC via RTS as before.

ROUTINE 4: SET AND RESET

Source code file name **USR4.SRC**
BASIC demo file name **USR4.BAS**

The following routine will set the screen colour according to the colour value of the parameter used in the USR command and will reset the colour back to the normal Atari blue when used without a parameter. Note that the routine can only be used with either one or no parameters and if used with more than one,

then a crash will result. The routine is called in the following two ways X=USR(1536,A) to set the screen colour and X=USR(1536) to reset it back to blue.

```
  LDX #148
  PLA
  BEQ RESET
  PLA
  PLA
  TAX
RESET
  STX 710
  RTS
```

The first instruction in this routine is to load the X-register with the number 148 using LDX (LoaD X-register) which is the default value for the normal Atari blue screen. The next instruction is the PLA which removes the odd byte from the stack and since this routine allows the use of only one or no parameters to be used in the USR command, the next instruction, the BEQ (Branch if EQual (to zero)) is used to determine in which mode the routine is being used. If the value was found to be zero, that is the USR call was made without a parameter, then the routine branches to the label RESET. The default colour value of 148 originally loaded into the X-register at the start of the routine is then stored in location 710 using STX (STore X-register) after which the routine is exited using RTS. If however the odd byte was not zero (in other words it was 1), that is the USR command was made with a parameter, then the parameter is removed from the stack using the two PLAs and its low-byte loaded into the accumulator. The colour value is then transferred into the X-register using TAX overwriting the default value of 148 originally put there. The value is then stored in location 710 using STX after which the routine is exited back to BASIC via RTS as before. Also note that PLA does not have to be the first instruction in the source code.

ROUTINE 5: RETURNING VALUES TO BASIC

Source code file names USR5A.SRC
USR5B.SRC
BASIC demo file names USR5A.BAS
USR5B.BAS

The variable used to action a machine code routine when using USR (X in these examples) can be used to return values to BASIC after the routine has been executed. The value loaded into the BASIC variable is determined by the two zero page locations 212 and 213 in the normal low/high 2-byte format. The following routine works in the opposite mode to the previous routines in that this routine randomly selects the colour value itself, changes the screen colour accordingly and returns the value to BASIC so that you know what the selected value was. The routine is executed using X=USR(1536) where the colour value is loaded into the variable 'X'. The routine may also be actioned using "PRINT USR(1536)" or "? USR(1536) which will print the selected colour value directly on screen.

Listing 1 **(USR5A.SRC)**

```
PLA
LDA 53770
STA 710
STA 212
LDA #0
STA 213
RTS
```

Listing 2 **(USR5B.SRC)**

```
PLA
STA 213
LDA 53770
STA 710
STA 212
RTS
```

In the first listing, the first instruction is the normal PLA which removes the odd byte from the stack. Next the processors accumulator is loaded with a random number from location 53770 (the random number generator) which is in the range of 0 to 255 inclusive. The random number selected is stored in the colour register 710 as before and also into location 212, the low-byte of the value for returning to BASIC. The accumulator is then zeroed by loading it with the number zero which is then stored in location 213, the high-byte of the value for returning to BASIC because the high-byte will always be zero. The routine is then exited using RTS as before.

The second listing shows a little trick that can be done by re-arranging the listing to reduce the length of the routine to save memory (2 bytes in this case). Because no parameters are used when calling this routine, the odd byte will automatically be zero which will be loaded into the accumulator by the PLA instruction. Since the accumulator contains a zero at this point, it makes good programming sense to take advantage of it and use it to zero location 213 so that "LDA #0" may be omitted from the listing and so reduce the length of the final routine.

Note that there is a bug in the Turbo BASIC compiler which prevents USR from returning the correct values to BASIC. If a program using this feature is to be compiled, then the two locations 212 and 213 could be PEEKed from BASIC in the normal way and then the two values combined using the usual low/high two byte calculation as follows....

```
X=PEEK(212)+256*PEEK(213)
```

A MEMORY MOVER ROUTINE

Source code file name USR6.SRC
BASIC demo file name USR6.BAS

To conclude this article, a very useful memory moving routine which works in a similar way to Turbo BASICs MOVE command. The routine will move up to 256 bytes of data from one address in memory to another. The routine may be used in conjunction with PMGs to move shape data into the player stripes to create smooth vertical movement as demonstrated in the BASIC demo program. The routine is executed using....

```
X=USR(1536,ADDR1,ADDR2,NUM)
```

Where NUM number of bytes is moved from address ADDR1 to address ADDR2.

```
  PLA
  PLA
  STA 204
  PLA
  STA 203
  PLA
  STA 206
  PLA
  STA 205
  PLA
  PLA
  STA 207
  LDY #0
LOOP
  LDA (203),Y
  STA (205),Y
  INY
  CPY 207
  BNE LOOP
  RTS
```

Note the necessary use of the zero page locations 203 to 206 for accessing the two addresses using indirect indexed addressing. Also note that location 207 is used to store the number of bytes to be moved, but this location need not reside in page zero, it may in fact reside anywhere in memory. ●

The CLASSIC PD ZONE

by Austin Hillman

FAREWELL TO FUTURA

THE FINAL ISSUES

This issue we present the concluding half of this examination of the FUTURA disk magazine which is now available from the Page 6 Library.

FUTURA SIXTEEN

Text articles include - ATARI 8-BIT NEWS - Other Atari 8-bit supporters. SOFTWARE SCENE - Kevin Cooke reviews Tube Baddies. DTP ATARI8 - Eric Bemrose continues his column on Atari 8-bit desktop publishing. This time he looks at Daisy-Dot II. ADDING A MOUSE - Part 5, Multi-Mouse & Board Games. HARDWARE WAREHOUSE - Atari DOS Disk Structure. THE ATARI 8-BIT BOOKSHELF - Booklist Part 6, S-Z, and Appendix A. SOFTWARE SCENE 2 - Kevin Cooke returns with a review of Demo Maker Update. VCS FUTURA - Cartridges by Atari (CX2601 - CX2699). THE BLACK AND RED - Introduction and News, Jaguar CD-ROM drive and game tips. Cannon Fodder reviewed by Michael Clatworthy. Raiden review and game tips by Michael Clatworthy. Tempest 2000 reviewed by Dan Baverstock.

The programs include *PING* - An extremely addictive game which can be played with an ST mouse or joystick. *ULTRA TRANSLATOR* - The ultimate translator for your XL/XE! It can even be used with 400/800 cartridges.

DOS WIZARD - Is a superb disk analyzer which must be loaded with a translator. Use Ultra Translator, simply press SELECT at the title screen and then load DOS Wizard. Please read the documentation first before using this powerful program. Using DOS Wizard you can gain a detailed analysis of single density disks, recover files mistakenly deleted, display complete sector maps or individual sectors, etc.

MAGNIFY - An impressive screen dump utility with extra features. Try it with the two pictures, *BOX* and *COVER*, which come with Ping.

FUTURA SEVENTEEN

Text articles include - ATARI 8-BIT NEWS - J.F. Software, Micro Discount. AMS95 REPORT - Kevin Cooke reports from the show of the year. THE ATARI 8-BIT BOOKSHELF - Atari Adventures, a book review by Kevin Cooke. DTP ATARI8 - Dot-Magic! TURBO BASIC FLYER - Ron Fetzer's programming column. ADDING A MOUSE - Part 6, GOE Demo. PROGRAMMER PROFILE - Lucasfilm's David Levine. THE BLACK AND RED - The Jaguar CD ROM package reviewed by Michael Clatworthy. Dino Dudes review and level codes. VCS FUTURA - Cartridges by Atari (CX26100 - CX26192). ATARI 8-BIT TRIVIA - Kevin Cooke issues a trivia challenge!

DESKTOP - from the French disk Cenacle News. Boot with BASIC and the desktop will auto-load. Use a joystick (or the arrow keys) to move the pointer to D.1 and press the joystick button (or Return). A list of all the files on the disk will appear. Select a file to load by moving the pointer over it and pressing the joystick button. This desktop will load both BASIC and machine code files. Not quite up the standard of Windows 95, but not bad for the humble 8-bit.

CHOPP THE ROBOT - A colourful, challenging arcade game from Cenacle News One. You must guide Chopp the Robot down the screen, rescue the stranded human and return to the mother ship. *ANAGRAM* - A utility which computes anagrams, of up to six letters. From Cenacle News Six. *OTHELLO BLITZ* - A version of the popular board game also from Cenacle News Six. Use a joystick to move the pointer and press the button to lay a piece. *BSA 73* - Version 2 of a BASIC (in all senses of the word) work simulation game written by Alan Hitchen, set in the days when we still had a motorcycle industry.

QUICKREF 1.2 - Analyses the structure of your BASIC programs. Docs supplied. *PROGRAM HELPER* - Slim down your BASIC program by converting constants into variables. Docs supplied. *FMT* - A very useful utility which allows you to format with or without DOS files. Works with the desktop. *FUJI 128* - Good demo. *PARAM, SPACEBAR, DECIMALS* and *GRLOADER* are programs described in the Turbo BASIC Flyer column.

LINE-UP - Comes from Chile and describes itself as the ultimate version of Tetris in the public domain, who am I to argue with that description. *THE BITTER REALITY MEGADEMO* - This is a 9-part extravaganza which really shows the power potential of a well progammed XL/XE.

FUTURA EIGHTEEN

Text articles include - ATARI 8-BIT NEWS - ACPC, Linefeed, The Page 6 ST Library. HARDWARE WAREHOUSE - Building a light pen. THE ATARI 8-BIT BOOKSHELF - Kevin Cooke reviews Itty Bitty Bytes of Space. NETWORK, TO THE WORLD - Communications article by Joe Hicswa. DTP ATARI8 - Daisy Dot III. PACMAN FOR THERAPY? - Kevin Cooke looks at modern medicine. THE TURBO BASIC FLYER - by Ron Fetzer. FRIENDLY VIEWS ON THE ST - by Les Wagar and Stuart Murray. ADDING A MOUSE - Part 7, The Brundles Editor. A SHORT HISTORY OF COMPUTERS - What came before your Atari 8-bit. CALLING ALL NOSTALGIA ENTHUSIASTS - Some light humour.

LISTER - This is a program for printing out your program listings complete with all of the special characters. There is an example listing

provided for you to play with. *QUICKDOS.BAS* - This program, from the old Atari User (Vol.3 No.2), alters DOS 2.5 to put DUP.SYS in the shadow RAM under the OS in XL/XE machines. After the first DOS call which runs from the disk as usual, all subsequent calls will run from the OS with no loss of memory as it is automatically saved.

DRAW800 - A simple but pleasing graphics demo showing the good ol' Atari 800. *PAGE6, PROGRAM1* and *PROGRAM2* are the programs to be used with the Turbo BASIC Flyer column.

THE CAVES OF CTULHI - Is a revised version of Robert de Letter's game which was published in NAU 70. *CTULHI'S REVENGE* - Is his all new sequel with loads of puzzle action!

FUTURA NINETEEN

Text articles include - ATARI 8-BIT NEWS - ACPC, UKAAUG. ADDING A MOUSE - Part 8, Noughts & Crosses. DTP ATARI8 - Digital Editor V3.6. SOFTWARE SCENE - Rambit Taskmaster. SOFTWARE SCENE 2 - Jawbreaker & Mousekattack. HANDY HINT - Ramdisk XL. A SHORT AND STUPID STORY - Light humour. VCS FUTURA - 2600 Audio Modification. PD SOFTWARE SCENE - Megablast. HARDWARE WAREHOUSE - Nintendo Controls on the Atari 8-bit. THE ATARI 8-BIT BOOKSHELF - Advanced Programming Techniques for your Atari. THE BLACK AND RED - Atari: The End?

NOUGHTS & CROSSES - New software from Kevin Cooke. See his column ADDING A MOUSE. *CARTOON SLIDESHOW* - Colourful artwork by Kevin Cooke. *KEYMASTER* - A keyboard enhancement utility, adds many functions. Full docs are included.

FONT MASTER - Allows fast printing of program listings, including special characters. There are lots of options available. *SIDEWAYS.DOC* is a test file for use with the sideways Syncalc print option. Two fonts, ROMAN and MODERN, are included for use with Options 3 and 4. *LIKEN MY ATARI* - A superb singalong music and graphics demo by Philip Price and Gary Gilbertson. *LABEL 720* - Allows you to read and write identity tags to your disks in the otherwise unused sector 720. *FINANCIAL CALCULATION PROGRAM* - Offers help with your investments and loans, works out many difficult calculations in a trice.

SADDLEMAN is a very enjoyable game from France. It was programmed for Atari France in 1985 as part of a nationwide contest in association with Levi's Jeans. It was never sold and is therefore in the public domain for us all to enjoy. There are five parts to the game. Use keys 1-5 to select a part. Press START to play and RESET to return to the main menu. Great fun!

FUTURA TWENTY

Text articles include - ATARI 8-BIT NEWS - Page 6, Telegames. DTP ATARI8 - Page Editor V3.3. SOFTWARE SCENE - Address Database and Envelope Printer reviewed by Kevin Cooke. VCS FUTURA - 2600 video Modification. THE BLACK AND RED - Jaguar news, reviews, tips and a look at Atari Entertainment.

CREATER.OBJ - A powerful menu utility. *MICRODOS.OBJ* - The classic menu utility for loading machine code files. *RD & RDF* - *PSI RAMdisk 2.03* and *PSI RDFormat 1.35* for use with SpartaDOS 3.2d. *FORMDOS1* & *FORMDOS2* - Format, DOS, DUP, RAMdisk all in one operation. A very useful utility by the late Alex Pignato. *SETUPDOS* - A tokenized BASIC routine to patch DOS 2.5 for two SD RAMdisks on a 256K XL.

FALCON DEMO - A top-notch demo by Hurek! Press the Spacebar at the scroller to progress to the main demo. *COLORS DEMO* - Kaleidoscope demo. *BOUNCE* - 24 demo boxes on one screen! *IMAGINE!* - A lightning fast graphics demo written in Action! *KEYCODE GETTER V3.0* - Keyboard codes utility. *AIR HOCKEY* - Fast sports game based on the old arcade favourite. Great fun in 2-player mode! *WEBMASTER* - Fun game, just eat the flies and keep the rival spiders away. Easy to play.

FUTURA TWENTYONE

The regular text articles come to a conclusion in this final issue. ATARI 8-BIT NEWS - Ol' Hackers and Cenacle still going strong. DTP ATARI8 - Parts 7 & 8, individual art and printing programs and series conclusion. HARDWARE WAREHOUSE - Re-inking printer ribbons. SOFTWARE SCENE - Demo Maker and Draw 7. VTOC - How it works. VCS FUTURA - 7800 video modification. THE BLACK AND RED - Jaguar news and Power Drive Rally review.

CREATE A FONT - A character set editor can be used with the 'Character Sets in an Instant' article. *DECODE* - A Mastermind-type quiz game by Frank Walters. *INSTANT CHARACTER SETS* - Article and a set of programs to convert alternative character sets into strings for instant loading when used with BASIC programs.

DISKLABEL 2.2 - Disk labeller plus printer driver. *COL80* - An 80-column file reader, with docs in 80 column format ready to read.

THE LAST WORD FROM STUART

Here's what Stuart said as his 'farewell' - "*So here we find ourselves at the final see-ya in the final Futura. I don't see this as a goodbye but more of a see-ya l-8-ter. I'm always going to be around on the 8-bit scene and will probably appear now and again with an article or review in one of the Atari publications. I'm looking forward to writing a series of articles on the Atari World Video Game Championships of 1982/83. It's about time I noted down all that happened behind the scenes!*

I'm really looking forward to getting stuck into 8-bit titles such as Karateka, Star Raiders II, Ultima IV, Infiltrator, etc., etc. There are so many titles I have purchased over the years that have hardly ever seen the inside of my disk drive! It is now time for me to enjoy them all.

All that is left to say is a final gigantic THANKS to everyone who has supported Futura over the years! You truly are an amazing group of people! Keep using that 8-bit and remember that together, WE are Atari! See-ya L-8-ter!"

SORTING AND SEARCHING

Daniel Yelland presents routines for you to use in your own programs

At first sight sorting and searching does not appear to be a very useful thing to do when programming, however a lot of programming problems can be solved with sorting and searching techniques.

A computer game is not the application you would immediately think of for using such techniques, but what about high score tables? In football manager games and adventure games searching for details is important also. More serious applications of sorting and searching could be in a database or accounting program. This article is about some of the techniques used when sorting and searching data.

Some search techniques require the data to be sorted which explains the relationship between the two processes. The examples given will all involve numbers but could equally well be used on alphanumeric data. Also unless stated otherwise the data is being sorted into ascending order.

SORTING

SELECT SORT - The most obvious sort algorithm is the one we ourselves use when sorting data. We look through the data to find the smallest element and place it in position 1, then we repeat the process for the next smallest element and so on until the data is sorted. This technique is called the Selection sort and an example program demonstrating this is given in SELECT.BAS. You will find the program on this issue's disk. This is how it works - given 5 numbers, say

15, 7, 3, 10, 9

we look through the list at each element in turn until we find the smallest. In this case it turns out to be 3, so

3 is put to the beginning of the list so the data now looks like this

3, 15, 7, 10, 9

We repeat the process, now ignoring the 3 as we know it is in order. The next smallest number is 7 and so that is placed in the list

just after the 3

3, 7, 15, 10, 9

The process is repeated until the list is sorted

3,7,9,15,10
3,7,9,10,15

This technique is the most obvious but it is not the fastest or most efficient, so other techniques have been tried and tested.

BUBBLE SORT - Most people have heard of the Bubble sort, it is probably the most famous of computing sort routines but it is also one of the worst. In this technique neighbouring pairs of elements in the list are compared and if the element higher in the list has a lower value than the element lower in the list the pair are swapped. The whole list is compared in this way with swaps taking place until the list is completely sorted. Usually a flag is set to report no swaps were made during a pass, which signals the list is sorted.

So with our original data set

15, 7, 3, 10, 9

First Pass

7, 15, 3, 10, 9 — 15 > 7 so swap
7, 3, 15, 10, 9 — 15 > 3 so swap
7, 3, 10, 15, 9 — 15 > 10 so swap
7, 3, 10, 9, 15 — 15 > 9 so swap

The list after first pass is

7, 3, 10, 9, 15

The largest value (15) has "bubbled" to the top which is where the technique gets it's name from.

Second Pass

7, 3, 10, 9, 15
3, 7, 10, 9, 15 — 7 > 3 so swap
3, 7, 10, 9, 15 — 7 < 10 so no swap
3, 7, 9, 10, 15 — 10 > 9 so swap
3, 7, 9, 10, 15 — 10 < 15 so no swap

List after second pass:-

3, 7, 9, 10, 15

In this case the list is sorted after two passes. It should be noted that 3 passes are made as the process doesn't terminate until there are no swaps. **BUBBLE.BAS** on this issue's disk is an example routine showing this technique.

INSERTION SORT - Another famous one, this technique is one of the most efficient but also one of the hardest to code. It is done the way people sort a hand of cards. For example consider the hand of cards dealt as - K 5 2 9 6

Sort

K 5 2 9 6
5 K 2 9 6
2 5 K 9 6
2 5 9 K 6
2 5 6 9 K

This is done by looking at the list from left to right and "correcting" elements which are out of order. On each pass we compare with the items on the left, until we find one larger, or reach the start of the list. The others "shuffle up" to make way for each moved element as in the example above. We insert the current item in the correct place and repeat this up until the last item. The program **INSERT.BAS** demonstrates this.

SHELL SORT - This routine could be said to be a variation on the Bubble sort. Invented by Donald Shell, in this method elements of a fixed gap apart are compared rather than adjacent elements as in the bubble sort. The

elements of the fixed gap apart are sorted in the same way as the bubble sort and then the gap is halved and the process is repeated. This is done until the gap is equal to 1 where a standard bubble sort then takes place.

The advantage of this routine over a standard bubble sort is that elements get to their correct position in the list quicker than in the bubble sort as elements travel further towards their correct position in each swap.

An example of this routine is in **SHELL.BAS**

PROGRAMMING THE TECHNIQUES

Since all of the data to be sorted is in lists the best data type to use to hold the elements is an array. Each element of the array holds an element of the list. Variables are used to hold the elements being compared and Loop structures are used to provide the iteration of the routines (e.g. continue until list sorted).

SEARCHING

LINEAR SEARCH - Again the most simple search routine is the one we ourselves use when searching for data. We look through the list until the element is found and stop when we either find the element or we have looked through the whole list.

So looking for 10 in the original list would result in:

Original list:- 15, 7, 3, 10, 9

item 1 (15) not equal to 10 so continue
item 2 (7) not equal to 10 so continue
item 3 (3) not equal to 10 so continue
item 4 (10) equal to 10 so stop
"Found" is returned.

If the whole list is searched and the element not found we stop the process and "Not Found" is returned. An example program of this is shown in **LINEAR.BAS**.

BINARY CHOP - This is a very efficient routine in that any element of 1000 elements can be found in just 10 passes. This routine however requires the data to be sorted first.

The routine works by using pointers which can be represented as variables in BASIC which point to the beginning of the list (e.g. array) and the end of the list. These will be referred to as the "start pointer" and "end pointer" respectively. Assuming the list is in order a middle value is taken from the list and compared against the search value. If it is equal to it the process terminates and "FOUND" is returned. If the middle value is less than the search value the starting pointer is set to the data item after the middle value in the list. If the middle value is greater than the search value the end pointer is set to the data item below the middle value in the list. After the pointers have been adjusted the process is repeated until either the element is found or the pointers coincide or cross one another (e.g. end pointer points to an element below the element start pointer points to.) **BINCHOP.BAS** illustrates this technique. The sort routine used to sort the data is the same one in SELECT.BAS.

There are many other sort and search techniques and the ones shown here are only a few moderately well known ones. Hopefully these routines will be of use in your own programs. ●

SPARTADOS

An overview and list of commands

compiled by M. Tomlin

SpartaDos is probably the most sophisticated disk operating system developed for the Atari 8-Bit system, and it knocks spots off Atari Dos 2.5. It supports multiple sub-directories (known as folders on the ST), full time/date stamping of files, full random access to any byte within a file, batch command files, hard disk access, and a whole host of more specialist features.

SpartaDos keeps most of it's code locked away under the operating system so you get more free memory for your programs. Even with the most powerful configuration, you will still have 32,501 free bytes available to Basic, and the smaller versions can leave as much as 36,176 available. These figures should be set against 32,274 with Dos 2.5 and 37,902 with no Dos at all!

GET A CLOCK!

If you can find one (new or second hand) use ICD's R-Time 8 battery backed-up clock cartridge and the correct time and date will always be available to be stamped on each new file created. The clock module plugs into the standard cartridge port and has a replacement socket on the top for another cartridge. It has been tried with Action!, Mac/65, Basic/XE, AtariWriter, Atari Assembler/Editor and so on - and it seems completely transparent to everything. There is even a Z: driver which allows you to access the clock directly from Basic.

US DOUBLER

Where SpartaDos really comes into its own is in conjunction with another ICD product - the US Doubler. This used to come in the form of two plug-in ICs for the 1050 drive which not only give it the ability to work in true double density but also speeds up the data transfer rate considerably. Once these chips are installed, your drive is supercharged. Upon booting your SpartaDos master disk the old bleep, bleep, bleep sound is replaced by a burst of rapid reading speed. Reading data from a disk is roughly four times faster than with a standard 1050 drive, but after allowing for seek time (the time taken to move the head around the disk) the average speed increase is nearer three times. The increase in speed of writing is less at roughly twice as fast. The other major feature of the US Doubler is to give access to true double density, giving 180k per disk as opposed to 90k in single or 130k in enhanced density. Double density actually uses 720 sectors per disk, as with single density, but each sector contains 256 bytes of data rather than 128. This does mean that you'll have to remember to format and write in standard mode if you wish to give a disk to somebody without a US Doubler,

SpartaDos Commands (vers. 3.2)

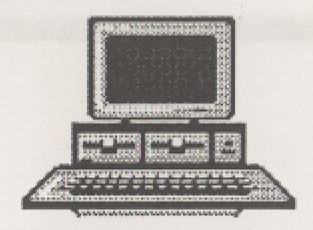
and comparison with DOS 2.5

Dos 2.5	SpartaDos	Function
A	DIRS	Disk directory, any drive (Dos 2.x format)
B	CAR	Go to cartridge (if present)
C	COPY/XCOPY	Copies file(s) (multiple drives)
D	ERASE	Delete file(s) from disk
E	RENAME	Rename file(s) on disk
F	PROTECT	Protect file(s) from accidental erasure
G	UNPROTECT	Remove erasure protection from file(s)
H	n/a	Write Dos file (handled during format by XINIT)
I	XINIT	Formats disk (see also AINIT)
J	DUPDSK	Duplicate whole disk
K	SAVE	Save binary file (see also APPEND)
L	LOAD	Load binary file (see also OFF LOAD)
M	RUN	Run machine code at given address
N	n/a	Create MEM.SAV (SpartaDos is always in memory)
O	XCOPY	Copy file(s) (single drive) (see also SPCOPY)
P	AINIT	Format (Single density Dos 2.0 mode)
	APPEND	Save binary file at end of exiting file
	AUTOBAT	Select batch file to run when Reset is pressed
	BASIC ON/OFF	Turns internal Basic on or off
	BOOT	Set filename to load when no Dos present on disk
	BYPASS	Modify hard disk drive access number
	CHKDSK	Give current disk statistics
	CHTD	Change time/date stamp on file(s)
	CHVOL	Change volume name of disk
	CREDIR	Makes new sub-directory
	CWD	Change default path details for current drive
	DATE	Set system date
	DELDIR	Delete sub-directory (must be empty)
	?DIR	Shows path to specified sub-directory
	DIR	Disk directory (Extended format: time/date/bytes)
	DUMP	Print file as Ascii + hex digits to screen
	KEY ON/OFF	Type-ahead buffer on or off
	LOCK/UNLOCK	Protect whole disk from write operations
	MDUMP	As for DUMP, but to print memory contents
	MEM	Show current Dos lomem/himem values
	MENU	Load menu system (may be set as default)
	OFF-LOAD	Binary load file, with offset
	PAUSE	Wait for a key to be pressed (in batch files)
	PORT	Change the RS-232 configuration
	PRINT	Echo screen output to another device like P: or C:
	PUTRUN	Add run address to binary file
	RD	Set up ramdisk (many configurations available)
	RPM	Test disk drive rotation speed
	RS232	Load RS-232 driver for 850 module/P:R: connection
	TIME	Set system time
	TDLINE	Load time/date header line routine
	TD ON/OFF	Turn time/date line on or off (requires TDLINE)
	TREE	Shows all sub-directories/files (alphabetical)
	TYPE	Shows Ascii file contents on screen
	VERIFY ON/OFF	Turns disk write on or off
	XDIV	Turn off I/O redirection (disable PRINT/batch mode)
	ZHAND	Activate Z: for time/date handling from Basic
	-filename	Execute batch command file (extender = .BAT)
	filename	Execute machine code file (extender = .COM)
	Dn:	Select new default drive number, n

but this is very simple since SpartaDos is intelligent enough to detect what type of disk it is using.

OVERVIEW

SpartaDos has so many features that it is impossible to describe them all in full, but the list of SpartaDos commands will give you some idea of its amazing power and flexibility when compared with Dos 2.5. I like the time/date stamping of files and the sub-directory facilities, and their ease of use from Basic and other languages. SpartaDos can also read, write and format Dos 2 disks automatically, so you need never worry about which type you currently have in the drive. It can even handle a hard disk if you happen to have one. If you can get hold of a copy (or even a photo copy) of the excellent manual all the better but as most people nowadays will only be able to get a copy without the manual I have included a list of most of the available commands which should help.



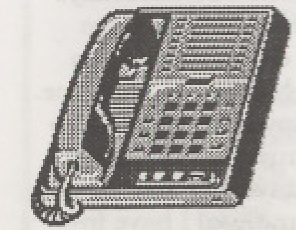
JOURNEY INTO CYBERSPACE

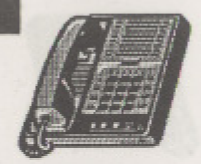
PRACTICAL USE OF THE INTERNET

John S Davison continues his exploration of the Internet

You can't have failed to notice that the Internet is now beginning to touch our everyday lives. It's not just mentioned in the mass media any more - they've actually begun to embrace it and directly encourage its use. For instance, many radio DJ's now regularly invite communication from listeners via e-mail while they're on the air; an increasing number of TV programmes now have their own World Wide Web addresses where additional information can be found; and many adverts on TV, in magazines and newspapers, and even on poster hoardings now quote the companies' WWW addresses along with their promotional messages.

But, there are still many people who view the Internet as a high technology toy used only by computer geeks who play with it "because it's there" or who use it purely for (dubious forms of) entertainment. This view generally stems from the fact that "non-believers" have no obvious practical use for it themselves. Such people need to use it "for real", i.e. with a specific purpose in mind, before they'll accept it for what it is - an incredibly rich source of information, entertainment, and increasingly, on-line commercial services (known variously as "e-business", "e-commerce", or even e-shopping"). So, over the next few issues I'll be looking at examples of practical use of the Internet to see how practical they really are.





OK, I know this isn't Atari specific, but there's no reason a suitably configured Atari couldn't be used in the ways described here.

TO AMERICA VIA CYBERSPACE

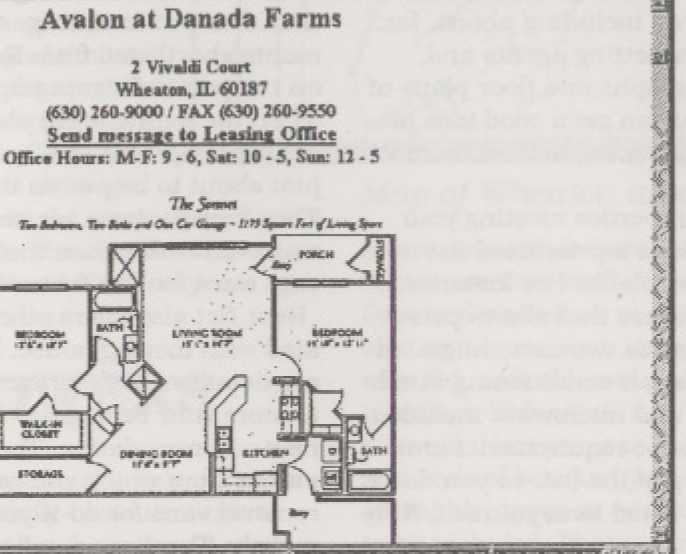
The practical use of the Internet was recently highlighted to us as a family following my elder son's decision to take up a job offer in the USA. Long time readers of NAU may remember John jnr - he wrote over 120 pieces for the magazine several years back while he was still at school. And thanks to this early experience (plus a lot of hard work) he's subsequently been able to build himself a very successful career in the magazine publishing industry. As we'll see, the Internet has figured prominently in his latest career move.

He recently accepted the post of Editor-In-Chief of Electronic Gaming Monthly, a hugely popular magazine for users of games consoles such as the Nintendo N64, Sony Playstation, and similar. The magazine is based near Chicago and much of the communication between John and his new employer was conducted via Internet e-mail, while he was still working in the UK. The time difference of 6 hours between London and Chicago makes telephone communication inconvenient, and e-mail provided the perfect solution. So, the Internet proved a real benefit to John, right from the start of negotiations.

FINDING A HOME

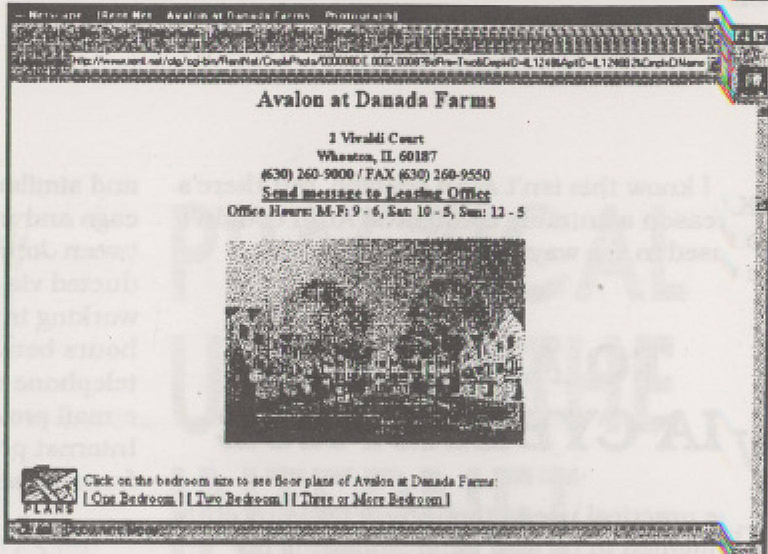
Once he'd accepted the job the hard work began. It's hassle enough moving to a new job in a different part of the UK, but when it's 5,000 miles away on a different continent the problems increase tenfold at least! The first challenge was to find somewhere to live. But how do you do that without spending weeks living in a hotel while trying to househunt and simultaneously attempting to get stuck into the new job? The answer - use the Internet as a househunting tool before you move to the new job.

John used a search engine to find the Web sites of companies that advertised properties for rent. The one he eventually settled on was called Rent.Net (http://www.rent.net, if you care to take a ☞



Floorplan of a property on Rent.Net





Picture of property development in Wheaton on Rent.Net

look). This lets you search for rental properties anywhere in America (all 50 states), Canada, and in certain other parts of the world too. You tell it the type of property you want, e.g. an unfurnished apartment, and it then steps you through a series of pages where you refine your requirements. You tell it which USA state you want tn live in (Illinois in John's case), and then which major city area (Chicago). It then presents you with a list of Chicago suburban areas. After selecting the suburbs of interest (Lombard and Wheaton in John's case, as his new company is based there) you tell it the size of property required (e.g. one bedroom, two bedroom, etc.) and the price range to look for. A list of available properties to suit your requirements and pocket then appears. Clicking on these brings up further information, including phone, fax, or e-mail details of the letting agents and, most usefully, photographs and floor plans of each property, so you can get a good idea of what they're like before going to view them for real.

If there are lots of properties meeting your criteria you can produce a prioritised list based on amenities available. For instance, you may want somewhere that allows pets (important to John as his two cats emigrated with him!); or that has air conditioning fitted; or has a dishwasher and microwave included. Properties meeting these requirements are then placed at the top of the list, so you don't have to search every listed item yourself. Nifty or what?

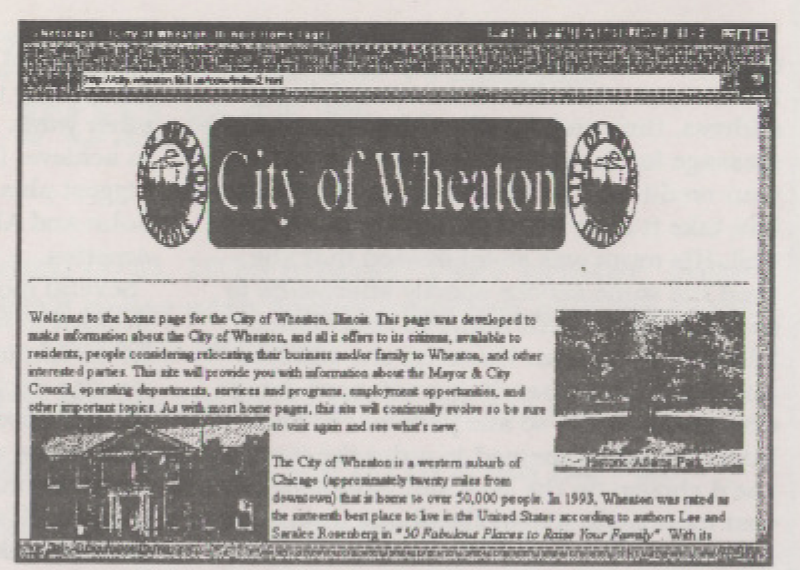
RESEARCHING THE LOCALITY

Obviously, you also want to know something about the area in which the available properties are located. Those on John's shortlist were in Wheaton, so by using a search engine to find references to Wheaton you can then click your way to the City of Wheaton home page (http://city.wheaton.lib.il.us/cow/index2.html). Through this and other links contained there you can find out virtually everything you need to know about Wheaton and what's currently going on there. It's regularly updated too.

John and his wife Ali (yes, he's married now!) then flew out to Chicago to view the apartments shortlisted from Rent.Net. They were on the point of signing up for one of them when the letting agent showed them a brand-new development in Wheaton that was only just about to be put on the Rent.Net Web site. They found a town house in this complex that was even better than their original choice, so they went for this instead.

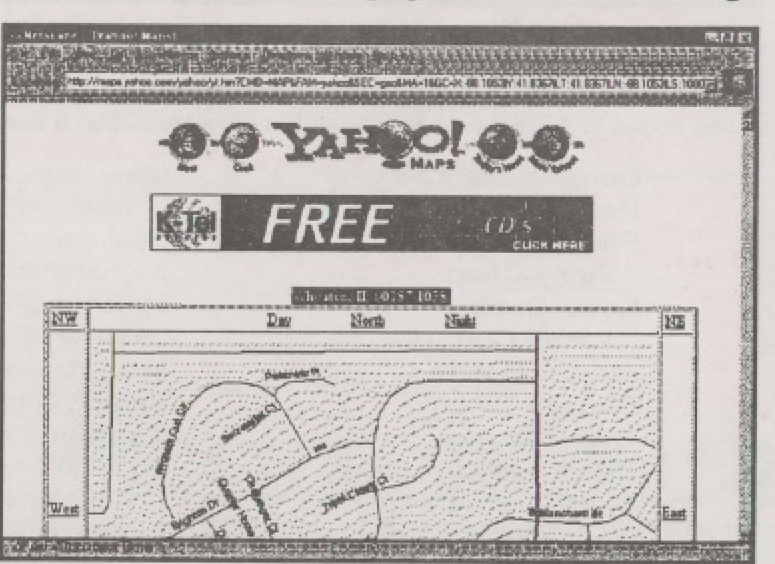
Rent.Net also offers other facilities associated with moving house. For instance, there are details of self storage units you can rent to store your belongings in while you sort out new accommodation. There are links to companies from which you can hire self-drive removal vans for do-it-yourself removals. There are details of local schools and child care services, cable TV hook-up companies, and insurance companies. If you don't have any furniture you can even rent some from here too. John and Ali's own furniture was being shipped from the UK by sea and took about six weeks to reach Chicago. They'd rented the town house well before the furniture delivery date, so faced the prospect of a month or more without a stick of furniture in the place. The solution was to take out a short term rental on a houseful of furniture and household appliances until their own stuff arrived.





City of Wheaton Home Page

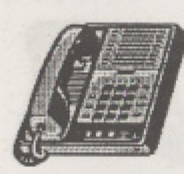
You can also find the location of the properties using an on-line map. There are maps on Rent.Net showing Chicago and surrounding areas, but there's a much better map facility available through Yahoo, at http://maps.yahoo.com. You key in the property's full postal address and a fully zoomable map of the area appears, with your selected address marked on it. You can zoom out until the map shows the whole USA or zoom in until you have a local streetmap of the immediate area around the specified address. If you then supply the address of your current location the system will provide driving directions from there to the property you want to view. It is rather like the AutoRoute program available for the Atari ST a few years ago.



Map of Wheaton area by Yahoo Maps

E-FLOWERS

Since John moved to Chicago we've kept in touch with him on a daily basis through e-mail. Phone calls cost far more than e-mail, so are reserved for weekends and special occasions. One such occasion was Mother's Day, and as well as phoning his mum John also sent her a large bouquet of flowers. He selected, ordered, and paid for these via the Interflora Web site (http://www.interflora.co.uk). Here you can choose a price range and style of bouquet you want to send, specifying the types or even the colours of flowers to be included. Photographs of suitable bou-



quets are displayed, from which you choose the one you want. You then enter delivery address, time and date for delivery, and a message for the accompanying card (plus your credit card details, of course), and Interflora take it from there. It works amazingly well. His mum was so impressed that she used the same service shortly afterwards to send flowers to HER mum, to cheer her up following an illness. However, she still has misgivings about sending credit card details over the Internet, so she phoned our local Interflora florist instead to order the item she'd chosen on the Web site. Annoyingly, it cost more to do it over the phone than via the Internet.

Well, I think the above is proof positive that the Internet does have practical applications. Admittedly it could all have been done in other ways, but would have taken far longer to achieve. Immediacy is one of the Internet's biggest plus points, and it certainly helped John and Ali to rapidly set up their new life in America.

Several months on from John's move we decided to take a holiday in the USA. Naturally, we wanted to go to Chicago to see him, but we also decided to visit friends in Texas whom we'd not seen for several years. You can probably guess what's coming next. Yes, we arranged most of the trip with the help of the Internet. It's a whole story in its own right, so we'll be looking at this in the next issue. ●

NAU Internet Contact List

The following NAU readers would welcome e-mail contact from other Atari users. If you'd like to be added to this list please drop an e-mail note to John S Davison at the address below.

Daniel Baverstock	dbaverstock@mistral.co.uk
Paul Bramley	p.bramley@student.qut.edu.au
Paul Carlson	paul.carlson@hn.se
Johnny Chan	jwchan@clara.net
Kevin Cooke	kj-cooke@wpg.uwe.ac.uk
Michael Current	mcurrent@carleton.edu
John S Davison	john_davison@compuserve.com
Damian Dixon	damian@tenet.co.uk
Gary Dundas	davadar@hotkey.net.au
Derek Fern	101755.2443@compuserve.com
Dean Garraghty	dgs@clara.net
Joel Goodwin	jgoodwin@wilco.co.uk
Paul Herbert	147378.97@swansea.ac.uk
Gordon Hooper	ua558@freenet.victoria.bc.ca
Fred Meijer	fmeijer@dsv.nl
Ann O'Driscoll	annod@iol.ie
Allan Palmer	100644.1040@compuserve.com
Paul Rixon	rixonp.railtrack@ems.rail.co.uk
Paulo A Rodrigues	nop25450@mail.telepac.pt
Brad Rogers	brad@pianosa.demon.co.uk
Nigel Turton	npturton@msn.com
Henning Wright	kofta@algonet.se
Daniel Yelland	yh182530@stmail.staffs.ac.uk
Bryan Zillwood	b.j.zillwood@exeter.ac.uk

contact ... contact ... contact ...

Only one notice for CONTACT this issue so Gary Partington gets a huge free advert!

COMPUTER HARDWARE FOR SALE

Atari 800XL computer (dual O/S), 1020 printer, 1027 printer, 1029 printer - all in good working order with power supplies and manuals - OFFERS INVITED

Modem cable - RS232 25 pin (F)/9pin (F) to 36 pin (M) - £5

RS232 'D' Connector Convertor 25 pin (F) to 9 pin (M) - £2

Printer Cable - Centronics - 32 pin (F) to 25 pin (M) - £5

5.25 inch computer disks - (55) (some new) - £10

5.25 inch disk drive cleaning disk kit - £5

Tel. Gary Partington on 01758 613524

FREE TO SUBSCRIBERS

The CONTACT column is free of charge to subscribers who wish to sell their equipment or contact other readers. Space is limited so we request that entries be kept as short as possible. Extremely long entries may be heavily edited or ignored. Send your CONTACT notice on a separate sheet of paper (not as part of a letter) to:

CONTACT, PAGE 6 PUBLISHING, P.O. BOX 54, STAFFORD, ST16 1DR

FOR SALE ... WANTED ... PEN PALS ... ADVICE ... HELP

PROGRAM LISTINGS

Certain program listings which are too long to include in the magazine may be obtained free of charge as printed listings to type in. All programs are, however, included on the Issue Disk which is available with each issue. Remember this disk also includes BONUS PROGRAMS which do not appear in the magazine. If you would like the type-in listings please write or telephone indicating which listings you require. Please note that there are not necessarily extra listings for every magazine.

Write to LISTINGS, NEW ATARI USER, P.O. BOX 54, STAFFORD, ST16 1DR or telephone 01785 241153